GRAPH BASED RECOMMENDER SYSTEM USING LINK PREDICTION

Synopsis of the PhD Thesis

Submitted by

Anshul Gupta

(Roll No 70821016008)

Department of Computer Engineering

Supervisor

Dr. Pravin Shrinath

submitted in partial fulfilment of the requirements of

DOCTOR OF PHILOSOPHY (Engineering)

MUKESH PATEL SCHOOL OF TECHNOLOGY MANAGEMENT & ENGINEERING
SVKM'sNARSEE MONJEE INSTITUTE OF MANAGEMENT STUDIES (Declared as Deemed-to-be University Under Section 3 of the UGC Act, 1956)
V. L. Mehta Road, Vile Parle (West)
MUMBAI –400056

April 2023

TABLE OF CONTENTS

List of Figures 4

List of Tables 5

Abstract 6

Chapter 1 Introduction 7

Chapter 2 Literature Review 8

2.1 Related Work 8

2.2 Research Gaps 11

Chapter 3 Objectives of the Work 12

Chapter 4 Contributions 13

4.1 Contribution 1 13

4.2 Contribution 2 13

4.3 Contribution 3 13

4.4 Contribution 4 14

Chapter 5 Mathematical Background 15

5.1 Introduction 15

5.2 Markov Models 15

5.3 Hidden Markov Models 15

5.4 Non-Negative Matrix Factorization 15

5.5 Affinity Propagation (AP) Clustering 15

5.6 Performance Evaluation Metrics 16

Chapter 6 A Recommender System Based on Collaborative Filtering, Graph Theory using HMM based Similarity Measures 17

6.1 Introduction 17

6.2 Problem Formulation 17

6.3 Proposed Methodology 19

6.4 Experimental Results 21

6.5 Conclusion 23

Chapter 7 Non-negative Matrix Factorization with Temporal Similarity Approach for Recommendation System 25

7.1 Introduction 25

7.2 Proposed Algorithm 25
7.3 Experimental Results 29
7.3.1 Scenario 1 30
7.3.2 Scenario 2 32
7.3.3 Scenario 3 32
7.3.4 Scenario 4 33
7.4 Conclusion 34
Chapter 8 Link Prediction based on Bipartite graph for Recommendation System using optimized SVD++ 35
8.1 Introduction 35
8.2 Proposed Link prediction-based Recommendation System 35
8.3 Experimentation and Result Analysis 37
8.4 Conclusion 38
Chapter 9 Conclusion 39
Chapter 10 Future Scope 40
Chapter 11 List of Publications 41
References 42

LIST OF FIGURES

FIGURE 6.2: FLOW CHART OF THE PROPOSED ALGORITHM. 19
FIGURE 7.1: BLOCK DIAGRAM OF PROPOSED RATING ESTIMATION SYSTEM. 26
FIGURE 7.2: ILLUSTRATION OF THE PROPOSED RATING ESTIMATION APPROACH THROUGH WNMF AND AP. 27
FIGURE 7.3: ILLUSTRATION OF THE PROPOSED RATING ESTIMATION APPROACH THROUGH WNMF AND AP. 28
FIGURE 7.4: ILLUSTRATION OF THE PROPOSED RATING ESTIMATION APPROACH THROUGH WNMF AND AP. 29
FIGURE 8.1: PROPOSED LINK PREDICTION BASED RECOMMENDATION SYSTEM. 36

LIST OF TABLES

TABLE 6-1: USERS PURCHASING BEHAVIOR IN TERMS OF MOVIE IDS. 18
TABLE 6-2: ALL POSSIBLE TRAIL FOR USER $u1$ GRAPH IN FIGURE 6.1. 18
TABLE 6-3: RESULTS COMPARISON BETWEEN SINGLE USER (W=1.0) VS CF (W=0.5) FOR DIFFERENT AGGREGATION WEIGHTS. 21
TABLE 6-4: RESULTS COMPARISON BETWEEN SINGLE USER ($w = 1.0$) VS CF ($w = 0.5$) FOR DIFFERENT RECOMMENDATION LENGTHS. 22
TABLE 6-5: RESULTS COMPARISON BETWEEN SINGLE USER ($w = 1.0$) VS CF ($w = 0.5$) FOR DIFFERENT LEARNING LENGTH. 22
TABLE67-6: RESULTS COMPARISON BETWEEN SINGLE USER ($w = 1.0$) VS CF ($w = 0.5$) FOR DIFFERENT GROUP SIZE. 23
TABLE 7-1: MEASURED PRECISION VALUES FOR SCENARIO 1. 31
TABLE 7-2: MEASURED RECALL VALUES FOR SCENARIO 1. 31
TABLE 7-4: MEASURED MAE VALUES FOR SCENARIO 1. 31
TABLE 7-5: MEASURED RMSE VALUES FOR SCENARIO 1. 32
TABLE 7-6: MEASURED MAE VALUES FOR SCENARIO 2. 32
TABLE 7-7: MEASURED MAE VALUES FOR MOVIELENS DATASET AT SUCCESSIVE TIME SEGMENTS FOR SCENARIO 3. 33
TABLE 7-8: MEASURED MAE VALUES FOR TA-FENG DATASET AT SUCCESSIVE TIME SEGMENTS FOR SCENARIO 3. 33
TABLE 7-9: HR@K (K = 10) PERFORMANCE COMPARISON FOR SCENARIO 4. 33
TABLE 8-10: NDCG@K (K = 10) PERFORMANCE COMPARISON FOR SCENARIO 4. 33
TABLE 8-1: EXPERIMENTATION RESULTS 37

ABSTRACT

Collaborative filtering (CF) is a widely used method in recommendation systems. In the CF, similar users' interests and preferences are analyzed and items that they might be interested in are recommended. Existing methods however calculate the coefficient of correlation to measure the correlation between users, which cannot measure sequential and temporal similarity among users which leads to inferior performance. In this work, we proposed three different graph based approaches to manage these problems properly.

The first approach approach achieves link prediction based on bipartite graph using optimized SVD++. The link prediction in approach is done with a bipartite network, and it is based on matrix factorization. The final linkages between users and items are predicted using the SVD++, which are subsequently optimized using the KNN technique. Finally, the performance of the proposed algorithms are evaluated using the MovieLens1M and Ta-Feng grocery datasets under different scenarios using several performance evaluation metrics such as Root F-Score, Mean Square Error (RMSE), Mean Absolute Error (MAE), etc., which validates the superior performance of the proposed algorithm over state-of-the-art algorithms.

The second approach tackles the problem using a graph-based approach that utilized the EMISSION Matrix to estimate the similarity between the users. The EMISSION Matrix used with the proposed algorithm is a part of HMM (Hidden Markov Model). In HMM the EMISSION matrix is used to estimate the possibility of emission of a particular symbol from a process. In our problem, we mapped the process with the user and the symbol with the selected item. Matching the EMISSION matrix of users' item selection process provides an improved similarity measure which also includes the sequential preferences. The proposed approach also overcomes the limitations of the previous algorithm which restricts the prediction domain to the user’s browsing history only.

The third approach considers a combined static and temporal similarity based CF is presented, which estimates the final ratings using a weighted sum of four independent ratings. The first rating is estimated through the similarity estimated using the Affinity Propagation (AP) applied to a reduced rank coefficient matrix generated by factorising the user-item rating matrix using Weighted Non-Negative Matrix Factorization (WNMF). The remaining three ratings are estimated by employing graph-based similarity measures on bipartite graphs made between the user and their static (age and occupation), and temporal (purchasing pattern) characteristics.

CHAPTER 1
INTRODUCTION

Information resources on the Internet are proliferating, making it more and more challenging for users to find what they need. Moreover, the rate of information growth far exceeds the capacity of people to process it, resulting in an overabundance of information [1]. Information overload caused by complex information in this era of competition is time-consuming. The recommendation systems suggest the most relevant item (product) to a user based on the user's purchasing history and the histories of other users showing similar behaviors. These predictions are based on similarities between products and the closeness of user preferences [2].

A website, for example, may give a plethora of options, but for those who lack the personal expertise or competence to make an informed decision, RSs are the best option. Examples include a system that recommends books based on user preferences. Amazon.com uses an RS to customize the online store for each individual consumer. Individuals and groups of people benefit from a variety of specialized recommendations since recommendations are often customized to each individual user.

In their simplest form, personalized recommendations are offered as ranked lists of items. In performing this ranking, RSs try to predict what the most suitable products or services are, based on the user's preferences and constraints. In order to complete such a computational task, RSs collect information from users regarding their preferences which are either explicitly expressed, e.g., as ratings for products or are inferred by interpreting the actions of the user. For instance, an RS may consider the navigation to a particular product page as an implicit sign of preference for the items shown on that page.

Personalized suggestions are presented in the form of a ranked list of things. In order to arrive at this ranking, RSs consider the preferences and limits of the users in order to make educated guesses about the best appropriate items and services. User preferences are either openly stated (e.g., product ratings) or inferred (e.g., by analyzing the user's actions) by RSs to perform a computational work like this one. If an RS sees a user navigate to a specific product page, they may interpret this as an implicit endorsement of the items on that page.

CHAPTER 2
LITERATURE REVIEW

2.1 Related Work

In this section, we review previous studies on recommender systems for e-commerce data, graph-based approaches for collaborative filtering, and the integration of user and item similarity in collaborative filtering algorithms. The Tapestry Project introduced recommendation systems (RS) in 1992 [14]. Personalized recommendations are provided by the practical recommender applications that help users deal with information overload by filtering masses of useless information [15].

The k-nearest neighbors is the typical collaborative filtering algorithm most widely applied. It is also called memory-based filtering [16]. In such a recommender system, one user is compared to k other users with similar tastes based on items they choose or evaluate, then recommended items that the k other users chose [17]. A fuzzy C-means clustering approach was proposed by Hamidreza Koohi et al [18]. Training sets consist of five different ratings. After that, clustering techniques such as K-means, SOM, and fuzzy clustering are used to create clusters for predicting a new user's rating and providing recommendations.

The two general types of recommendation systems are memory-based and model-based, depending on how they are implemented. The memory-based method generates recommendations directly from the database, while the model-based method generates recommendations from transaction data [2]. The memory-based method, which accesses the database directly, is adaptive to changes in data but is time-consuming according to the data size. For the model-based method, its computing time is constant regardless of the size of the data, but it is not adaptive to changes in the data [19].

The hybrid model proposed in [6] combines the use of genomic tags to identify movies with a content-based filtering concept. The method uses Principal Component Analysis (PCA) in conjunction with Pearson Correlation to reduce redundant tags, reducing computation complexity.

A transaction-based recommender system (TBRS) named hierarchical attentive transaction embedding (HATE) model aims to predict the next item by modeling dependencies in transactional data was presented in [20]. In their proposal the set of recent transactions is treated as the inter-transaction context while the already-chosen items in the current transaction form the intra-transaction context.

Deep learning enjoys a massive hype at the moment. Recently, deep learning has been revolutionizing the recommendation architectures dramatically and brings more opportunities to improve the performance of recommender [21]. Considering the capabilities of deep learning a DeepFM model which combines the power of factorization machines (FM) [22] for

recommendation and deep learning for feature learning in a new neural network architecture was presented in [23]. Compared to the latest Wide & Deep model from Google [24], DeepFM has a shared input to its "wide" and "deep" parts, with no need of feature engineering besides raw features.

Another approach based on improvement over traditional FM named Dual Input aware Factorization Machines (DIFMs) was presented in [22]. The DIFM was aim to learn more flexible and informative representations of a given feature according to different input instances. It can adaptively reweight the original feature representations at the bit-wise and vector-wise levels simultaneously. Furthermore, DIFMs strategically integrate various components including Multi-Head Self-Attention, Residual Networks and DNNs into a unified end-to-end model. A deep learning based RS named as CoupledCF for non-IID coupling relations between users and items was proposed in [25]. The CoupledCF first learns the explicit user-item couplings w.r.t. user attributes and item attributes by a CNN-based user-item coupling learning network, then builds a deep CF (DeepCF) model to learn the implicit user-item couplings, and finally integrates the learned explicit user item couplings with DeepCF to systematically represent user, item and user-item couplings.

Although the deep learning seems promising for normal cases the standard machine learning models also performs reasonably [26]. Hence the use of deep neural networks are justified only when there is a huge amount of complexity or when there is a large number of training instances [21].

A distributed collaborative coupling learning system which enables differential privacy was presented in [27], which defends against the adversary who has gained full knowledge of the training mechanism and the access to the model trained collaboratively. It also addresses the privacy-utility tradeoff by a provable tight sensitivity bound.

The matrix factorization model is one of the most powerful and is commonly used for movie recommender systems [28] It uses a user-item matrix where users are represented by rows while movies are represented by columns.

Matrix factorization have become potent solution to reduce the dimension of data, extract latent features of the data and mitigate sparsity [29]. They are used widely in recommenders owing to their characteristics. The matrix factorization techniques used in recommenders include Singular Value Decomposition (SVD) [30], Principal Component Analysis (PCA) [31], Probabilistic Matrix Factorization (PMF) [32], [33] and Non-negative Matrix Factorization (NMF) [34]. These factorization models can be formulated as optimization problem with objective function and constraints. In KDD Cup 2011 [35] recommender system competitions, it was observed that alternating least squares (ALS) [36] and stochastic gradient descent (SGD) [37] have attracted much attention and are widely used for matrix factorization [38].

In matrix factorization problem (shown in Eq. 1) ALS alternatively switches between updating W and updating H while fixing the other factor. Although the time complexity per iteration is $O(|\Omega|k^2 + (m+n)k^3)$, [39] shows that ALS is well suited for parallelization. ALS can be easily parallelized when entire W and H can fit in the main memory of one computer. However, it is hard to be scaled up to very large-scale recommender systems when W or H cannot fit in the memory of a single machine [38].

As mentioned in [28], SGD has become one of the most popular methods for matrix factorization in recommender systems due to its efficiency and simple implementation. The time complexity per iteration of SGD is $O(|\Omega|k)$, which is lower than ALS. However, as compared to ALS, SGD needs more iterations to obtain a good enough model, and its performance is sensitive to the choice of the learning rate. Furthermore, unlike ALS, parallelization of SGD is challenging.

In matrix factorization problem (shown in Eq. 1) the total time complexity of GCD to update W and H once is $O(|\Omega|k^2)$ operations per iteration, which is the same complexity as ALS. ALS can be easily parallelized when entire W and H can fit in the main memory of one computer. However, it is hard to be scaled up to very large-scale recommender systems when W or H cannot fit in the memory of a single machine.

The traditional matrix factorization based approaches share the limitation in that the interaction between user embedding and item embedding is only weakly enforced by fitting the given individual rating value, which may lose potentially useful information. To overcome this problem a novel Augmented Generalized Matrix Factorization (AGMF) approach that is able to incorporate the historical interaction information of users and items for learning effective representations of users and items was presented in [40]. Unlike [25] in [41] a non-deep learning based approach using novel generic coupled matrix factorization (CMF) model for non-IID coupling relations between users and items was proposed. Such couplings integrate the intra-coupled interactions within an attribute and inter-coupled interactions among different attributes.

One of the most critical factors that greatly affect the performance of CF is the computation of similarities between users. Cosine (COS), Pearson correlation coefficient (PCC), adjusted cosine measure (ACOS) and Spearman's rank correlation (SRC) are the generic traditional measures for similarity computation [42], [43]. However literature has shown that the traditional measures do not properly utilize the user preferences (ratings data); especially when the available user item rating matrix is sparse or ratings data are not sufficient. Therefore several new similarity measures to improve the performance of a RS were introduced local and global user similarity measure. Such as new similarity measure called JMSD [44] by joining Jaccard and Mean squared-difference (MSD). Mean–Jaccard–Difference (MJD) [45] combination of six similarity measures. At that point, the neural network is utilized to tune the weights of each similarity measure. In proposed another similarity measure that chooses neighbors dynamically for each different target item was presented [46]. In [47] an iterative message passing procedure for similarity updating was utilized.

Another common approach to a recommender system is based on the graph [48]–[50]. Most of the data in RS has essentially a graph structure where the objects here considered including users, items, attributes, etc. are tightly connected with each other and influence each other via various relations [51]. In [52] heterogeneous edge embedding for friend recommendation system on social networks was presented. This method effectively exploits edge heterogeneity in multi-graphs to mine network representation. Another graph based approach for representation learning for attributed multiplex heterogeneous network is presented in [53]. In [50] an approach to learning user-item relatedness from knowledge graphs for top-N item recommendation was presented. The graph learning-based recommender systems (GLRS) have been extensively studied [54].

2.2 Research Gaps

RS are developing rapidly. Although great progress has been achieved as illustrated in the last section, some challenges still remain not well resolved and thus require more efforts. This work addresses the following research gaps associated with RS:

1. The existing techniques lack the **incorporation of temporal context**. Hence, they fail to reflect the temporal shifts in user behaviors in their recommendations.
2. Existing methods calculate the coefficient of correlation to measure the correlation between users, which **cannot measure sequential similarity among users**. Also these algorithms restricts the prediction domain to the user's browsing history only.
3. The complexity of the similarity measurement in graph-based techniques increases exponentially with the size of the dataset and the search depth, which causes **large memory and processing power requirements and sluggish performance**.
4. The important aspect of user similarity-based CF is to estimate the similarities among users. Traditionally, the similarity among the users is calculated using the user-item matrix. However, using the **higher-dimensional user-item matrix leads to longer processing times and large memory requirements**. Furthermore, the user-item matrix in most cases is sparse, which causes the poor similarity measure performance.

CHAPTER 3
OBJECTIVES OF THE WORK

To fulfill the research gaps following list of objectives were set for this work:

1. The incorporation of temporal context, to **reflect the temporal shifts in user behaviors** in recommendations.
2. **The development of an improved similarity measure, which also includes the sequential preferences.** Also overcomes the limitations of the previous algorithm which restricts the prediction domain to the user's browsing history only.
3. To develop an **efficient method for measuring graph similarity that requires less memory and processing power**.
4. To develop an **efficient matrix representation** for the large user-item matrix **without losing the information and mitigate the impact of sparsity** in similarity measurement.

CHAPTER 4
CONTRIBUTIONS

After finding the research gaps and the limitations of the existing techniques, and based on these findings, contributed by proposing the following techniques:

4.1 Contribution 1

It surveys the techniques and tactics utilized in RS and reviews it using collaborative filtering and similarity measures. Performance metrics used by collaborative recommender systems to detect similarity among users or items and to evaluate RS performance are also examined. Different social networking and e-commerce sites' techniques and existing recommender systems' difficulties and limits are also evaluated.

With this contribution following objective were achieved:

1. A review of the existing techniques was performed to build the ground for the new research.
2. An understanding of working of different approaches used for RS have been developed.
3. The knowledge of different datasets evaluation metrics and mathematical models were also developed.

4.2 Contribution 2

This research contributes to the enhancement of recommender engine suggestions using link prediction. Machine learning-based approaches are used in this work to present a solution to the problem of sparse data, scalability, and improved prediction of future relationships between users and items connected in a graphical fashion. The top predicted links are recommended to users in a recommendation system, which is analogous to the classification problem. Matrix factorization using SVD++ and the K-nearest neighbor (KNN) technique are used to identify the top 'k' predictions. Matrix factorization is used as a basis for a general-purpose link prediction model, and optimize the recommendations using KNN.

With this contribution following objective were achieved:

1. To develop an efficient matrix representation for the large user-item matrix without losing the information and mitigate the impact of sparsity in similarity measurement.

4.3 Contribution 3

A graph-based approach that utilizes the EMISSION Matrix to estimate the similarity between the users has been proposed. The EMISSION Matrix used with the proposed algorithm is a part of

HMM (Hidden Markov Model). In HMM the EMISSION matrix is used to estimate the possibility of emission of a particular symbol from a process. In our problem, we mapped the process with the user and the symbol with the selected item. Matching the EMISSION matrix of users' item selection process provides an improved similarity measure which also includes the sequential preferences.

With this contribution following objective were achieved:

1. An improved similarity measure which also includes the sequential preferences.
2. Overcomes the limitations of the previous algorithm which restricts the prediction domain to the user's browsing history only.
3. An efficient method for measuring graph similarity that requires less memory and processing power.

4.4 Contribution 4

In this work a combined static and temporal similarity based CF is presented, which estimates the final ratings using a weighted sum of four independent ratings. The first rating is estimated through the similarity estimated using the Affinity Propagation (AP) applied to a reduced rank coefficient matrix generated by factorising the user-item rating matrix using Weighted Non-Negative Matrix Factorization (WNMF). The remaining three ratings are estimated by employing graph-based similarity measures on bipartite graphs made between the user and their static (age and occupation), and temporal (purchasing pattern) characteristics.

With this contribution following objective were achieved:

1. The incorporation of temporal context, to reflect the temporal shifts in user behaviors in recommendations.
2. To develop an efficient matrix representation for the large user-item matrix without losing the information and mitigate the impact of sparsity in similarity measurement.

CHAPTER 5
MATHEMATICAL BACKGROUND

5.1 Introduction

Since this work involves the integration of multiple engineering methods for feature extraction, signal representation, optimization, and classification, in this chapter these various techniques are explained.

5.2 Markov Models

Markov models are utilized to prepare and perceive sequential data, like, stock market, temperature variation, time series, and other sequential data. In a Markov model, every perception in the data sequence relies on past components in the sequence.

5.3 Hidden Markov Models

Despite its effectiveness, the Markov model possesses limited applicability in many applications. Therefore to enhance its applicability, an extension of the Markov model was developed and named as Hidden Markov Model (HMM). As its name suggests it deals with the process whose states are hidden (not known) and the states can only be probabilistically modeled to relate with the emitted symbols (observations).

5.4 Non-Negative Matrix Factorization

The non-negative matrix factorization is a process of decomposing a matrix containing non-negative data into two smaller rank matrices [65]. In general terms, this is an NMF problem, which can be stated as follows:

Let $\boldsymbol{X} \in \mathbb{R}^{m \times n}$ represents the given non-negative matrix, the objective is to find $\boldsymbol{W} \in \mathbb{R}^{m \times k}$, and $\boldsymbol{H} \in \mathbb{R}^{k \times n}$, where k, is chosen to be $k < min\{m, n\}$, such that:

$$\min_{W,H} f(\boldsymbol{W}, \boldsymbol{H}) = \left||\boldsymbol{X} - \boldsymbol{W}\boldsymbol{H}|\right|_F^2 \tag{5.1}$$

Where, $\boldsymbol{W}$ *and* $\boldsymbol{H}$ are the non-negative factors of $\boldsymbol{X}$, which contain the basis vectors and their coefficients respectively, and $\left||\cdot|\right|_F$ represents the Frobenius norm..

5.5 Affinity Propagation (AP) Clustering

Affinity propagation uses message passing between data points to form the clusters. This technique differs from the traditional clustering techniques such as K-Means or K-Medoids, which

require the number of clusters to be declared before starting the process. Instead, the AP finds the number of clusters automatically [67].

To form the clusters, the AP depends upon two types of messages, known as "responsibility" and "availability". The responsibility ((*i*,)) is accumulated information sent from data point i to exemplar k. This represents the relative suitability of candidate exemplar k for data point i with respect to all other candidate exemplars and is calculated as:

$$r(i,k) \leftarrow s(i,k) - \max_{k' s.t. k' \neq k} \{a(i,k') + s(i,k')\} \tag{5.3}$$

Where $s(i,k)$ and $(a(i,k))$ denotes similarity and availability, respectively.

5.6 Performance Evaluation Metrics

In this work several performance measures were used but important measures are listed below-

Exact Sequential Hit Ratio (ESHR): if the N item sequence is predicted by the recommendation system and the user's actual purchasing sequence is given as $\{a_i\}^N{}_{i=1}$ and $\{b_i\}^N{}_{i=1}$ respectively, then the Exact Sequential Hit Ratio is calculated as:

$$ESHR = \frac{\sum_{i=1}^{N} f(a_i, b_i)}{N}, where, f(a_i, b_i) = \begin{cases} 1, & a_i = b_i \\ 0, & a_i \neq b_i \end{cases} \tag{5.7}$$

The ESHR measures that how accurately the predicted sequence matches with exact purchasing, or the accuracy of the recommender system in predicting the exact purchasing order of items.

Hit Ratio (HR): The HR measures how many items in the predicted sequence matches with exact purchasing in any order. The Hit Ratio is calculated as:

$$HR = \frac{|\{a_i\}^N{}_{i=1} \cap \{b_i\}^N{}_{i=1}|}{N} \tag{5.8}$$

Accuracy: it is the most simple performance measure and it is simply a ratio of correctly predicted recommendations to the total recommendations.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{5.9}$$

Precision: It is the fraction of items predicted by the recommender system that is actually purchased by the user.

$$Precision = \frac{TP}{TP + FP} \tag{5.10}$$

CHAPTER 6
A RECOMMENDER SYSTEM BASED ON COLLABORATIVE FILTERING, GRAPH THEORY USING HMM BASED SIMILARITY MEASURES

6.1 Introduction

Collaborative filtering (CF) is a widely used method in recommendation systems. In the CF, similar users' interests and preferences are analyzed and items that they might be interested in are recommended. Existing methods however calculate the coefficient of correlation to measure the correlation between users, which cannot measure sequential similarity among users. In this section, we proposed a graph-based approach that utilized the EMISSION Matrix to estimate the similarity between the users. The EMISSION Matrix used with the proposed algorithm is a part of HMM (Hidden Markov Model). In HMM the EMISSION matrix is used to estimate the possibility of emission of a particular symbol from a process. In our problem, we mapped the process with the user and the symbol with the selected item. Matching the EMISSION matrix of users' item selection process provides an improved similarity measure which also includes the sequential preferences. The proposed approach also overcomes the limitations of the previous algorithm which restricts the prediction domain to the user's browsing history only.

6.2 Problem Formulation

Before going to the explanation of the proposed methodology, let us first discuss the problem undertaken. This research addresses the problem related to the movie recommendation system. Where the recommendation system has to predict the most probable choice (movie) of a user at any given instant of time.

Table 6.1 presents the movies watched by 5 users over 3 consecutive events as provided by the MovieLens dataset. Now RS have to predict the movie choice of each user for the event t_4. If the choice is predicted for each user only through the user's own purchasing history, there exist only three recommendations. This is further useless as users have already purchased them and not going to purchase again. Therefore it is needed to provide the choice from the purchasing histories of other similar users (Collaborative Filtering). Even in this condition, there exist only 12 choices, considering the total 3900 movie titles in the dataset these are negligible. So the problem can be more acceptable if instead of movie titles the movie genres are considered. Because each movie in the MovieLens dataset is classified under 18 genres. Table 6.2 shows the genre-based representation of Table 6.1.

Table 6-1: Users purchasing behavior in terms of movie IDs.

	Events			
Users	t_1	t_2	t_3	t_4
u_1	1193	661	914	?
u_2	1357	3068	1537	?
u_3	3421	1641	648	?
u_4	2987	2333	1173	?
u_5	3468	1210	2951	?

Table 6.2, shows that out of 18, 14 choices are available for the recommendation, also unlike movies titles the similar genre of movies (with a different title) can be purchased again, therefore the user's own history will have better participation with this approach.

The only problem that seems with this approach is multiple genres for the same movie. This may cause problems during the estimation of the future choice because when the process will be modeled as HMM it will represent the simultaneous emission of multiple different symbols. To deal with this problem a directed temporal tree graph-based approach is utilized. A directed tree representation branches the tree at the vertices (nodes) having multiple genres and creates a different trail for each genre of that node. The temporal tree graphs for users u_1 and u_2 is shown in Fig. 6.1 and Table 6-1 list the all possible trails between starting node to ending nodes for u_1.

Looking at the graph of user u_1, for each node at the event t_3, we can estimate the most probable symbol (genre) that may link at the event t_4. Now at the event t_4 the symbol with the highest aggregated probability can be recommended.

To extend the model for applying collaborative filtering (CF), we firstly required to find similar users. There are several approaches available to find the similarities between the graphs. However, traversing through every path in the graph to measure similarity is a complex and time-consuming task that grows exponentially with the node numbers. Therefore in this work, a probabilistic approach has been adopted.

Table 6-2: All possible trail for User u_1 graph in Figure 6.1.

	Events			
Trail	t_1	t_2	t_3	t_4
1	8	3	14	?
2	8	3	16	?
3	8	4	14	?
4	8	4	16	?
5	8	12	14	?
6	8	12	16	?

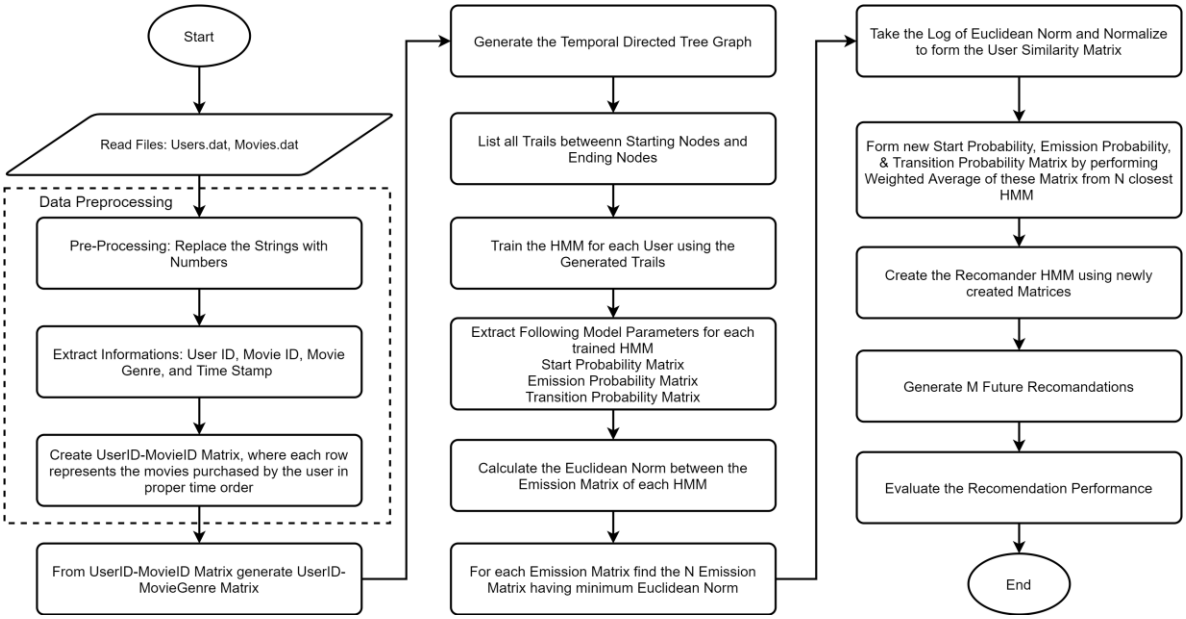

Start
Read Files: Users.dat, Movies.dat
Data Preprocessing
Pre-Processing: Replace the Strings with Numbers
Extract Informations: User ID, Movie ID, Movie Genre, and Time Stamp
Create UserID-MovieID Matrix, where each row represents the movies purchased by the user in proper time order
From UserID-MovieID Matrix generate UserID-MovieGenre Matrix
Generate the Temporal Directed Tree Graph
List all Trails betweenn Starting Nodes and Ending Nodes
Train the HMM for each User using the Generated Trails
Extract Following Model Parameters for each trained HMM
Start Probability Matrix
Emission Probability Matrix
Transition Probability Matrix
Calculate the Euclidean Norm between the Emission Matrix of each HMM
For each Emission Matrix find the N Emission Matrix having minimum Euclidean Norm
Take the Log of Euclidean Norm and Normalize to form the User Similarity Matrix
Form new Start Probability, Emission Probability, & Transition Probability Matrix by performing Weighted Average of these Matrix from N closest HMM
Create the Recomander HMM using newly created Matrices
Generate M Future Recomandations
Evaluate the Recomendation Performance
End

Figure 6.1: Flow Chart of the Proposed Algorithm.

The proposed probabilistic approach estimates the probabilities for each symbol to move on to every available symbol at the next event. The probabilistic approach requires traversing only through the single edges of the graph. Since in this work, each user is modeled as HMM, this probabilistic matrix also serves as an emission matrix used in HMM.

Once similar users are found the next task is to collaborate the information from these users to improve the recommendation performance. In the proposed approach each user is modeled as HMM, which stores all the information in the form of an initial state matrix, state transition matrix, and emission matrix. Hence to collaborate the information from similar users a new collaborative HMM is generated using the modified initial state, transition, and emission matrix. These modified matrices are generated by taking a weighted average of these matrices from similar users HMM. The weight depends upon the measured similarity.

6.3 Proposed Methodology

The flow chart of the proposed algorithm is presented in Figure 6.2 and this section presents a detailed explanation of the proposed algorithm in reference to the flow chart.

In the first step, the data files Users.dat and Movies.dat are processed and the string values are replaced by the numeric values to make the processing faster. Then the information is grouped based on userIDs.

Let the title of the total movie available on the server can be presented as $M = \{m_1, m_2, \dots, m_{N^M}\}$ where N^M is the total number of movies titles available on the server, and the Genres of each movie is presented as $G = \{g_1, g_2, \dots, g_{N^M}\}$, where $\forall i, g_i \subseteq O = \{o_1, o_2, \dots, o_{N^G}\}$,

where O represents the set of Genres and N^G is the total number of Genres (in MovieLens Dataset it contains 18 unique genres). $U = \{u_1, u_2, \dots, u_{N^U}\}$, represents the list of the users who have already purchased some movies, where N^U is the total number of users.

Let for any given time duration the movie purchasing pattern of the user u_i is defined as p_i where $p_i \in P = \{p_1, p_2, \dots, p_{N^U}\}$. where P stores the movie purchasing patterns of all users and p_i is the movie purchasing pattern of the i^{th} user.

The movie purchasing pattern of i^{th} user is expressed as the collection of movies $p_i^m = \{m_{i1}, m_{i2}, \dots, m_{iN^{M_i}}\}, m_{ij} \in M$, where N^{M_i} is the total number of movies purchased by the i^{th} user in a given duration. This move purchasing pattern is then converted to Genres pattern (p_i^g) by replacing each element of p_i by corresponding g_i in the set G. The $p_i^g = \{g_{m_{i1}}, g_{m_{i2}}, \dots, g_{m_{iN^{M_i}}}\}$, where, $\forall g_j^i \in G \wedge \forall g_j^i \subseteq O = \{o_1, o_2, \dots, o_{N^G}\}$, the elements of p_i^g may contain more than one value, and is used to generate a temporal directed tree graph. Through the generated graph is then utilized to generate all the trails sequences. The generated sequences are used to train the HMM using the equation described in section below, the training process estimates the transition (A_i) and emission matrices (B_i) for the i^{th} user. Ones the B_i is calculated for each user, the distance between the user i and j is calculated using the equation (6.1).

$$d_{ij} = \|B_i - B_j\|_2 where, 1 \le i, j \le N^U \ \wedge \ i \ne j \tag{6.1}$$

Where, $\|\cdot\|_2$, represents $L_2 - Norm$ operation. For each user only the N^S users having minimum d_{ij} are selected as similar users. The HMM parameters of similar users are combined to create the HMM for the recommender system. Let $U_i^S = \{u_{i1}, u_{i2}, \dots, u_{iN_s}\}, \forall j : u_{ij} \ne u_i$, and $D_i^S = \{d_{i1}, d_{i2}, \dots, d_{iN_s}\}$, contains the similar users and their distances with user u_i respectively. To create the HMM for i^{th} (or u_i) user recommender system the initial probability matrix (Π^i), state transition matrix (A^i) and emission probability matrix (B^i) are calculated as follows:

$$\Pi^i = 0.5\Pi_i + 0.5\left(\sum_{j=1}^{N_s} w_{ij} \times \Pi_j\right), where \ \forall \Pi_j \in \Pi_i^S \tag{6.2}$$

$$\mathrm{A}^i = 0.5\mathrm{A}_i + 0.5\left(\sum_{j=1}^{N_s} w_{ij} \times \mathrm{A}_j\right), where \ \forall \mathrm{A}_j \in \mathrm{A}_i^S \tag{6.3}$$

$$\mathrm{B}^i = 0.5\mathrm{B}_i + 0.5\left(\sum_{j=1}^{N_s} w_{ij} \times \mathrm{B}_j\right), where \ \forall \mathrm{B}_j \in \mathrm{B}_i^S \tag{6.4}$$

Where, Π_i^S, A_i^S, and B_i^S contains the initial probability, state transition, and emission matrix of the HMMs of users similar to u_i. While w_{ij} is the weight multiplier calculated by dividing each

element of D_i^S by maximum value element of D_i^S and then it is normalized in logarithmic scale as follows:

$$w'_{ij} = \log_e\left(\frac{d_{ij}}{d_{max}^{iS}}\right) : d_{ij} \in D_i^S \wedge d_{max}^{iS} = \max_{d \in D_i^S} d \tag{6.5}$$

$$w_{ij} = \frac{w'_{ij}}{\sum_{j=1}^{N^S}(w'_{ij})} \tag{6.6}$$

The matrices Π^i, A^i, and B^i are finally scaled to make the aggregate probability equals one. These matrices are then used to model the recommendation HMM.

6.4 Experimental Results

In addition to the preprocessing of a data set, the grouping of the movies accessed by the user is performed based on their UserIDs. To evaluate the performance of the recommendation system six different measurements named Exact Sequential Hit Ratio (ESHR), Hit Ratio (HR), Accuracy, Precision, Recall, and F-Measure are utilized. The results for the different model configurations are presented through Table 6-3 to Table 6-6.

Table 6-3 shows the performance of the proposed system for different user weight configurations. The user weight in the proposed CF recommendation system defines the contribution of different users in the same group (i.e. similar users). It can be seen from Table 6-3

Table 6-3: Results comparison between single user (w=1.0) vs CF (w=0.5) for different aggregation weights.

Here $\boldsymbol{w}$, represents the main user weight, while $\mathbf{1} - \boldsymbol{w}$, represents the group users' weight during aggregation to generate recommendation model.	**Model Configuration**	
	HMM States	**3**
	Number of Users	**20**
	History Learning Length	**5**
	Similar Users Group Size	**3**
	Recommendation Length	**3**
	Averaging Loop Counts	**5**

Weight (w)	**ES-Hit Ratio (%)**	**Hit Ratio (%)**	**Accuracy**	**Precision**	**Recall**	**F1**
0.1	20.23	43.00	72.07	28.98	22.26	25.15
0.2	30.21	43.67	72.25	29.29	21.65	24.89
0.3	35.79	43.67	72.12	29.35	22.47	25.43
0.4	39.95	46.00	72.29	30.58	24.32	27.06
0.5	39.75	45.00	72.16	29.77	22.88	25.86
0.6	34.90	44.67	72.07	30.50	24.74	27.31
0.7	29.56	41.33	71.90	29.58	23.50	26.17
0.8	24.17	39.67	71.37	28.26	22.68	25.16

Table 6-4: Results comparison between single user ($w = 1.0$) vs CF ($w = 0.5$) for different recommendation lengths.

Here $\boldsymbol{w}$, represents the main user weight, while $\boldsymbol{1 - w}$, represents the group users weight during aggregation to generate recommendation model.	**Model Configuration**	
	HMM States	**3**
	Number of Users	**20**
	History Learning Length	**5**
	Similar Users Group Size	**3**
	Averaging Loop Counts	**5**

Max. Recom.	**ES-Hit Ratio (%)**		**Hit Ratio (%)**		**Accuracy**		**Precision**		**Recall**		**F1**	
	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$
2	20.92	23.85	28.50	29.50	78.93	77.78	20.60	23.50	09.81	15.49	13.24	18.67
3	20.93	27.51	36.67	42.67	74.27	72.64	27.87	29.11	13.61	20.41	18.24	23.96
4	23.50	28.72	54.00	51.24	70.45	68.38	33.42	32.64	16.88	23.44	22.42	27.27
5	26.06	32.36	65.40	58.59	68.82	66.39	36.04	34.78	19.25	27.31	25.09	30.59

that for the lower values for the $w = 0.1$ the ES-Hit ratio remains at the lowest level (20.23%) because of the lower influence of the main user on the recommendation estimation. However, when the w reaches 0.5 the ES-Ratio attains 39.75%, achieves a significant relative improvement of 96.5% which is also the maximum ES-Ratio achieved, because at this point a proper balance between main user and similar users influence is attained. The further increase in the $w = 0.8$ reduces the system performance as the system relies more on the main user only and fails to gain diverse information from similar users. Table 6-4 also depicts that other measurements also show

Table 6-5: Results comparison between single user ($w = 1.0$) vs CF ($w = 0.5$) for different learning length.

Here $\boldsymbol{w}$, represents the main user weight, while $\boldsymbol{1 - w}$, represents the group users' weight during aggregation to generate recommendation model.	**Model Configuration**	
	HMM States	**3**
	Number of Users	**20**
	Similar Users Group Size	**3**
	Recommendation Length	**3**
	Averaging Loop Counts	**5**

Learning Length	**ES-Hit Ratio (%)**		**Hit Ratio (%)**		**Accuracy**		**Precision**		**Recall**		**F1**	
	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$
5	20.93	27.51	36.67	42.67	74.27	72.64	27.87	29.11	13.61	20.41	18.24	23.96
6	33.40	39.29	56.99	55.67	74.31	72.50	33.93	32.94	15.38	21.92	21.15	26.31
7	37.23	37.39	56.00	50.33	74.57	72.49	33.66	32.86	16.63	24.55	22.22	28.08
8	37.01	38.57	48.00	44.67	75.69	74.03	31.19	29.55	20.68	23.90	24.75	26.43

Table66-6: Results comparison between single user ($w = 1.0$) vs CF ($w = 0.5$) for different group size.

Here $\boldsymbol{w}$, represents the main user weight, while $\boldsymbol{1 - w}$, represents the group users weight during aggregation to generate recommendation model.	**Model Configuration**	
	HMM States	**3**
	Number of Users	**20**
	History Learning Length	**5**
	Recommendation Length	**3**
	Averaging Loop Counts	**5**

Group Size	**ES-Hit Ratio (%)**		**Hit Ratio (%)**		**Accuracy**		**Precision**		**Recall**		**F1**	
	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$	$\boldsymbol{w = 1.0}$	$\boldsymbol{w = 0.5}$
3	20.93	27.51	36.67	42.67	74.27	72.64	27.87	29.11	13.61	20.41	18.24	23.96
4	29.79	36.68	44.33	47.00	72.34	72.29	30.31	31.01	23.50	24.94	26.45	27.64
5	29.41	30.63	45.00	50.33	72.12	72.49	31.10	32.86	25.97	24.55	28.28	28.08

similar trend, however with lower variations as these measures do not consider the temporal information. Table 6-4 shows the performance comparison of the proposed system for $w = 1.0$ (i.e. single user) and $w = 0.5$ (i.e. CF) under varying recommendation lengths.

The results depict that for $w = 0.5$ the ES-Hit Ratio improves significantly with the length of recommendation. However, the HIT Ratio for $w = 0.5$ increases initially but decreases for higher recommendation length because of the difficulty in predicting long sequences in exact order. The F-Measure for the $w = 0.5$ provides better values than $w = 1.0$ for all recommendation lengths.

The effect of different group sizes on the performance of the system is presented in Table 6-5. These results show that for $w = 0.5$, and group size of 3 the proposed algorithm achieves an ES-Hit Ratio of 27.51% which is equivalent to 31.4% relative improvement over $w = 1.0$. However, for the group size of 5, it decreases by 4%. This behavior of the system is caused by the increase in model diversity for limited recommendation length. A similar performance is also noticed for F-Measures. Though the group size does not affect the Hit-Ratio performance, which always shows a relative improvement of 16%.

Finally, Table 6-6 evaluates the impact of learning history length on performance. The result shows that increasing the learning length decreases the performance. This decrease in performance depicts that user behavior at the current time depends only upon the most recent past activities, and not on the whole history. This gives an insight into users' behavior for the recommendation system.

6.5 Conclusion

This approach presented a temporal graph-based algorithm, which defines users' similarity through HMM parameters. The key features of the proposed algorithm are, i) it adopts a temporal graph-based algorithm to extract important information, ii) Similarity-based weighted averages

make the system more resilient to error-prone weight estimation which results in better collaboration of information, iii) the proposed method reduces the complexity of the similarity measurements, unlike graph-based which has an exponentially increasing number of branches. All these features lead to an overall performance improvement in terms of different evaluation measurements.

CHAPTER 7
NON-NEGATIVE MATRIX FACTORIZATION WITH TEMPORAL SIMILARITY APPROACH FOR RECOMMENDATION SYSTEM

7.1 Introduction

Users' similarity plays a crucial role in the Collaborative Filtering (CF) based Recommendation Systems (RS). The CF uses a user-item matrix to estimate this similarity. However, the user-item matrix based similarity performs poorly during cold-start and temporal variations in users' behavior. To overcome this limitation in this section, a combined static and temporal similarity based CF is presented, which estimates the final ratings using a weighted sum of four independent ratings. The first rating is estimated through the similarity estimated using the Affinity Propagation (AP) applied to a reduced rank coefficient matrix generated by factorising the user-item rating matrix using Weighted Non-Negative Matrix Factorization (WNMF). The remaining three ratings are estimated by employing graph-based similarity measures on bipartite graphs made between the user and their static (age and occupation), and temporal (purchasing pattern) characteristics. Finally, the performance of the proposed algorithm is evaluated using the MovieLens1M and Ta-Feng grocery datasets under different scenarios. The results show that the proposed approach not only outperforms the traditional techniques but also outperforms the Google Wide & Deep and CoupledCF models by 22.52%, 2.06%, and 11.26%, 2.75% for Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG), respectively.

7.2 Proposed Algorithm

The basic blocks of the proposed approach are presented in Fig. 7.1, which can be described by the following steps:

Step 1: Completing the User-Item Rating matrix (Fig. 7.2): Let $X \in \mathbb{R}^{m \times n}$ be the incomplete user-item rating matrix, where m and n represent the total number of items and users, respectively. The estimation of the missing ratings is done using the WNMF as presented in Section-III (a). Let the X^{est} represents the complete matrix, then using Eq. (7.2) this matrix can be factored into two lower rank matrices such that $X^{est} \approx WH$, where $W \in \mathbb{R}^{m \times k}$ and $H \in \mathbb{R}^{k \times n}$ matrices contain basis vectors and coefficients, respectively.

Step 2: Grouping the Similar Users: In the proposed work, similar users are grouped by multiple criteria as follows:

1. Users Rating Behavior.
2. Users Age Group.

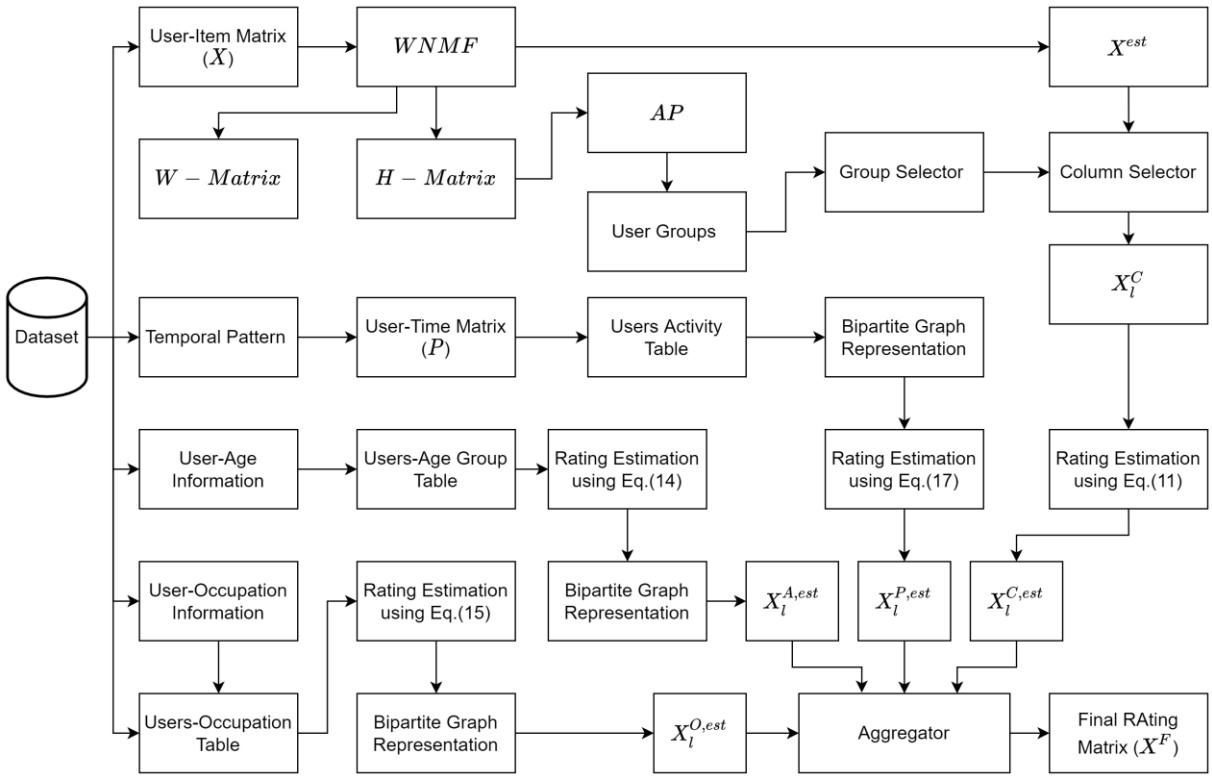


Figure 7.1: Block Diagram of Proposed Rating Estimation System.

3. Users Occupations.
4. Users Temporal Purchasing Pattern.

Step 3: Users Grouping Based on Rating Behavior (Fig. 7.3): In this approach, the similarity of the users is estimated using their rating similarity, or how they have rated the item so far. For this purpose, the matrix X^{est} can be used. However, due to the very large vector size, such estimation could require more memory and processing time. Also, the sparse distribution of information over large features in matrix X^{est} could lead to improper measurements. A better solution to this problem is the utilization of matrix H, described in Step 1.

Given that the column of the matrix H represents the compressed rating behavior of the respective user, similar users can be grouped using the Affinity Propagation (AP) clustering. Let the AP divide the users into N^C groups, and the user-item rating matrix for l^{th} group is given by X_l^C, then the missing rating for item j by a user k of this group is estimated using the following equation:

$$X_l^{C,est}(j,k) = \sum_{i=1, i\neq k}^{b:X_l^C \in \mathbb{R}^{a\times b}} w_{i,k} \times X_l^C(j,i) \tag{7.1}$$

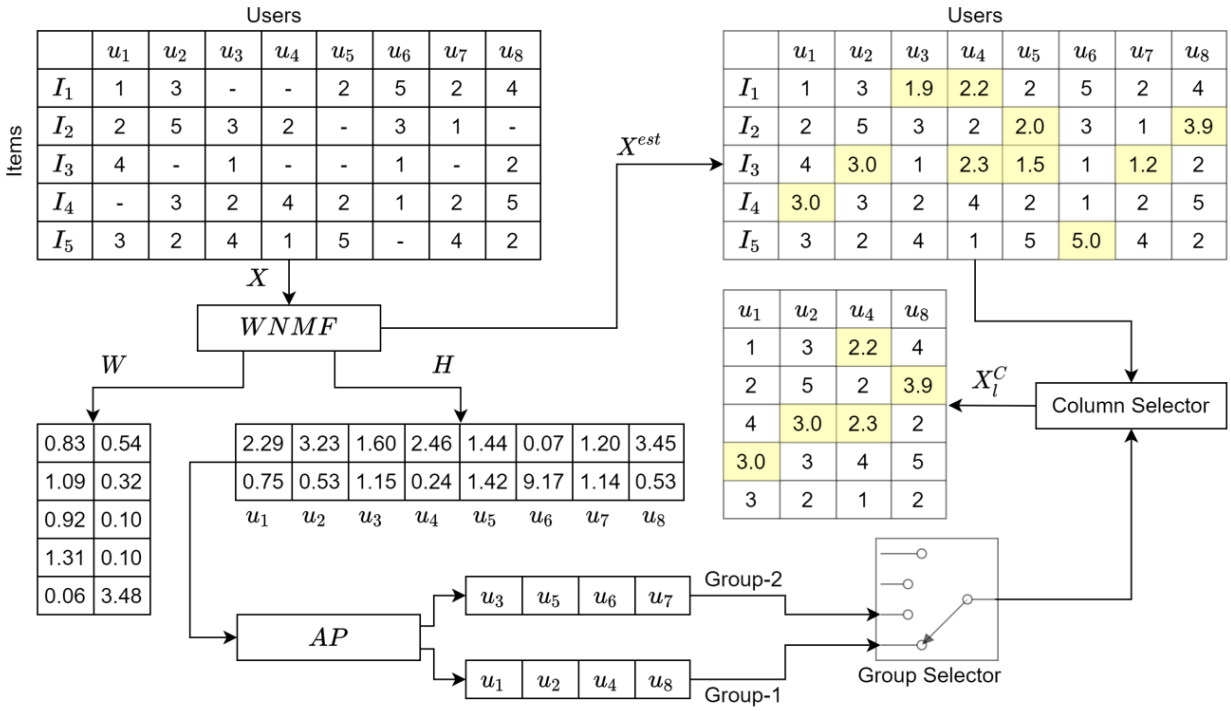


Figure 7.2: Illustration of the Proposed Rating Estimation Approach through WNMF and AP.

Where $w_{i,k}$ is given as:

$$w_{i,k} = \frac{1}{N-1}\left(1 - \frac{d_{i,k}}{\sum_{i=1}^{b:X_l^C \in \mathbb{R}^{a\times b}} d_{i,k}}\right) \tag{7.2}$$

Where $d_{i,k}$ is the Euclidean distance between column i and k of matrix X_l^C and given as:

$$d_{i,k} = \sqrt{\sum_{j=1}^{b:X_l^C \in \mathbb{R}^{a\times b}} \left\| X_l^C(j,i) - X_l^C(j,k) \right\|^2} \tag{7.3}$$

Step 4: Users Grouping Based on Age Group: The matrix X^{est} is divided into N^A groups where N^A denotes the total numbers of age groups defined in users' information. Let the users-item rating matrix of users belong to l^{th} group is given by X_l^A, then the missing rating for item j by a user k of this group. The estimated rating is the average rating for j^{th} item by the users in the group and is calculated using the following equation:

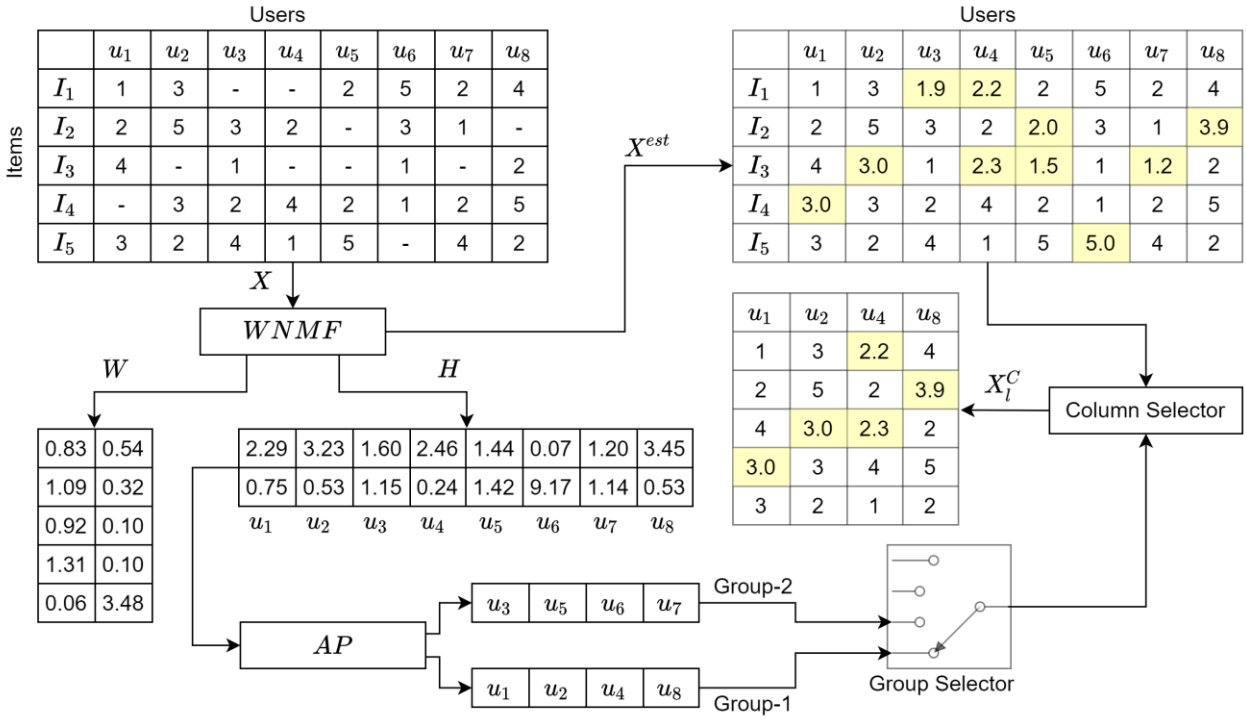


Figure 7.3: Illustration of the Proposed Rating Estimation Approach through WNMF and AP.

$$X_l^{A,est}(j,k) = \frac{1}{b}\left(\sum_{i=1,i\neq k}^{b} X_l^A(j,i)\right), where\ b: X_l^A \in \mathbb{R}^{a\times b} \tag{7.4}$$

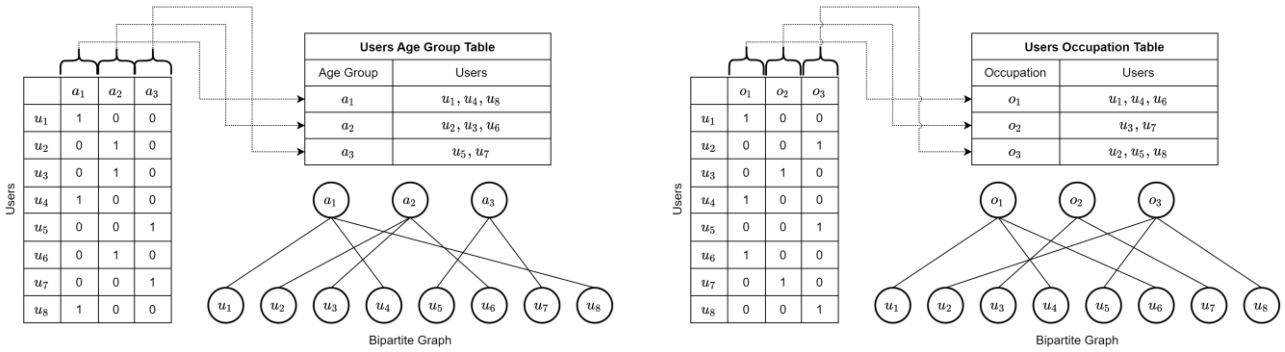
Step 5: Users Grouping Based on Occupation: This is also similar to age group and given as:

$$X_l^{O,est}(j,k) = \frac{1}{b}\left(\sum_{i=1,i\neq k}^{b} X_l^O(j,i)\right), where\ b: X_l^O \in \mathbb{R}^{a\times b} \tag{7.5}$$

Step 6: Users Grouping Based on Temporal Purchasing Pattern: To estimate this, first a matrix P whose rows represent the users and columns represent the time is created. The matrix elements are filled by either '1' or '0' according to the following condition:

$$P(i,j) = \begin{cases} 1, & if\ user\ u_i\ purchase\ at\ t_j \\ 0, & Otherwise \end{cases} \tag{7.6}$$

Based on a predefined constant (time segment duration, T_d.) the successive columns of the matrix P are measured. This process is repeated for another constant (sampling windows duration, T_w.). Using this process, a new table is formed where all users active in a given segment are grouped. The resulting table is then converted into a bipartite graph, and the similarity between



(a) (b)

Figure 7.4: Illustration of the Proposed Rating Estimation Approach through WNMF and AP.

users is calculated using cosine similarity. Let the user-item rating matrix for the users who appear in the same sampling window be given by, X^P, then the missing rating for item j by a user k of this group is estimated using the following equation:

$$X^{P,est}(j,k) = \sum_{i=1, i\neq k}^{b: X^P \in \mathbb{R}^{a\times b}} CS_{i,k} \times X^P(j,i) \tag{7.7}$$

Where $CS_{i,k}$ is the cosine similarity between user i and j.

Step 7: Aggregating the Ratings: up to Step-6, four different estimations of the missing ratings have been made. To get the final rating (X^F) all these ratings are aggregated using the following equation:

$$X^F(j,k) = \omega_C \times X_l^{C,est}(i,j) + \omega_A \times X_l^{A,est}(i,j) + \omega_O \times X_l^{O,est}(i,j) + \omega_P \times X^{P,est}(i,j) \tag{7.8}$$

Subject to:

$$\omega_C + \omega_A + \omega_O + \omega_P = 1$$

Where ω_C, ω_A, ω_O, and ω_P are the weights assigned to the cluster, age, occupation, and temporal rating estimators respectively.

7.3 Experimental Results

Several datasets are available for the performance evaluation of RS, such as MovieLens [57], Ta-Feng [58], Book-Crossing [59], Netflix Prize [60], Amazon Review Data [61], etc. In this work, however, only the MovieLens and Ta-Feng datasets were used to test how well the proposed recommendation algorithm worked. This is because only these datasets have features like the user s age and occupation and location, which are needed for the proposed algorithm to estimate static

similarity. The MovieLens datasets are available in different sizes (from several GBs to a few MBs). Originally released in 1998, this dataset was created to understand movie preferences. In this work, we adopted Movie- Lens1M variant, which has 1 million transactions from 6,040 users and 3,952 items. The user's purchasing behaviour is stored as tuples of ⟨ user, item, rating, timestamp⟩. Files users.dat and movies.dat, respectively, contain detailed information about the user and movies. The Ta-Feng dataset contains numerous baskets of purchased items from a grocery store, where each basket encapsulates the items purchased by one user in a period of time. This dataset contains 817,741 transactions belonging to 32,266 users and 23,812 items. The user's purchasing behavior is stored as tuples of ⟨transaction dt, customer id, age group, pin code, product subclass, product id, amount, asset, sales price ⟩ in a single file. The tuples of ⟨ time stamp, user id, user age, user occupation, movie id, movie rating ⟩ features were used for the MovieLens dataset, whereas tuples of ⟨ transaction dt, customer id, age group, pin code, product id, product purchased ⟩ features were used for the Ta-Feng dataset. Since the Ta-Feng dataset does not contain an occupation feature, we used pin code to estimate the static similarity. Lastly, four different scenarios were created to test and compare how well the proposed algorithm worked.

7.3.1 Scenario 1

In this scenario, performance was evaluated in terms of precision, recall, f-measure, MAE, and RMSE. Furthermore, a comparison with some traditional algorithms such as K-Means, AP (Affinity Propagation), FCM (Fuzzy C-Means) and WNMF (Weighted Non-Negative Matrix Factorization) was also performed. The experiments were performed for top-k recommendations where k varies from 5 to 50, and the results for different evaluation metrics are presented in tabulated form (from Table 7-1 to Table 7-5). The weights ω_C, ω_A, ω_O and ω_P were set to 0.5, 0.15, 0.15, and 0.2. The evaluation results for precision (Table 7-1) show that the proposed algorithm gives the best results for all top-K lengths on both datasets, also the precision gradually decreases for higher values of K. When compared for relative improvement, the proposed algorithm achieves the highest improvements of 10.56% and 19.85% and the lowest improvements of 4.15% and 5.67% for the MovieLens1M and Ta-Feng datasets, respectively. The results for recall (Table 7-2) show that the proposed algorithm performs best for all values of K on both the datasets except K = 5 where WMNF performs best for the MovieLens1M dataset. The recall also gradually increases for higher values of K. The highest achieved improvements for recall are 10.3% and 12.94% for MovieLens1M and Ta-Feng datasets, respectively. Similar to precision and recall, the proposed algorithm also achieves the highest f-score, which increases with K. The proposed technique also outperforms others for MAE (Table 7-4) and RMSE (Table 7-5) evaluations, but its performance decreases for higher values of K. The proposed technique achieves the highest MAE and RMSE improvement of 33.90% and 25.33%, respectively, for K = 10, for the MovieLens1M dataset, whereas for the Ta-Feng dataset the improvements are 25.33% and 22.72% respectively for K = 10. These results show that the proposed algorithm provides the best overall performance when compared with traditional algorithms.

Table 7-1: Measured precision values for scenario 1.

K	Precision									
	K-Means		FCM		AP		WNMF		Proposed	
	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng
5	0.343	0.331	0.396	0.355	0.410	0.388	0.406	0.381	0.427	0.410
10	0.321	0.305	0.365	0.342	0.383	0.367	0.362	0.348	0.404	0.392
20	0.288	0.256	0.320	0.301	0.341	0.327	0.331	0.310	0.377	0.361
30	0.239	0.210	0.298	0.289	0.322	0.298	0.320	0.285	0.355	0.345
40	0.220	0.192	0.282	0.271	0.305	0.273	0.295	0.263	0.337	0.326
50	0.211	0.184	0.273	0.262	0.292	0.257	0.286	0.245	0.321	0.314

Table 7-2: Measured recall values for scenario 1.

K	Precision									
	K-Means		FCM		AP		WNMF		Proposed	
	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng
5	0.053	0.052	0.071	0.06	0.073	0.065	0.082	0.067	0.076	0.075
10	0.103	0.098	0.124	0.11	0.136	0.122	0.132	0.11	0.144	0.134
20	0.168	0.156	0.236	0.201	0.248	0.21	0.25	0.224	0.269	0.253
30	0.279	0.265	0.323	0.308	0.346	0.312	0.366	0.332	0.381	0.363
40	0.356	0.327	0.425	0.401	0.436	0.414	0.414	0.396	0.481	0.452
50	0.419	0.402	0.488	0.456	0.521	0.502	0.485	0.435	0.574	0.532

Table 7-3: Measured F-Score values for scenario 1.

K	Precision									
	K-Means		FCM		AP		WNMF		Proposed	
	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng
5	0.018	0.015	0.028	0.023	0.03	0.026	0.033	0.036	0.032	0.035
10	0.033	0.026	0.045	0.041	0.052	0.046	0.048	0.045	0.058	0.061
20	0.048	0.042	0.075	0.068	0.086	0.072	0.083	0.082	0.101	0.098
30	0.066	0.056	0.096	0.084	0.111	0.11	0.117	0.112	0.135	0.132
40	0.078	0.068	0.119	0.107	0.132	0.124	0.122	0.123	0.162	0.156
50	0.088	0.078	0.133	0.112	0.152	0.142	0.138	0.124	0.184	0.174

Table 7-3: Measured MAE values for scenario 1.

K	Precision									
	K-Means		FCM		AP		WNMF		Proposed	
	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng
5	1.83	1.98	1.82	1.94	1.65	1.88	1.72	1.89	1.19	1.25
10	1.74	1.82	1.77	1.85	1.82	1.76	1.98	1.9	1.15	1.24
20	1.6	1.74	1.63	1.78	2.08	1.93	1.87	1.95	1.2	1.29
30	1.6	1.62	1.57	1.65	2.12	2.24	1.82	1.92	1.28	1.31
40	1.53	1.56	1.54	1.42	2.21	2.47	1.78	1.83	1.35	1.33
50	1.5	1.45	1.5	1.48	2.29	2.35	1.75	1.8	1.41	1.38

Table 7-4: Measured RMSE values for scenario 1.

K	Precision									
	K-Means		FCM		AP		WNMF		Proposed	
	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng
5	2.29	2.43	2.31	2.43	2.03	2.12	2.02	2.14	1.62	1.64
10	2.25	2.35	2.27	4.41	2.21	2.34	2.37	2.2	1.65	1.7
20	2.16	2.26	2.18	2.21	2.13	2.25	2.11	2.15	1.66	1.76
30	2.14	2.24	2.15	2.34	2.07	2.16	2.02	2.16	1.75	1.81
40	2.14	2.32	2.15	2.44	2.06	2.19	2.28	2.25	1.8	1.92
50	2.13	2.26	2.13	2.34	2.04	2.15	2.25	2.34	1.82	1.95

7.3.2 Scenario 2

In this scenario, the algorithms were evaluated for the cold-start condition. For cold start evaluation, there is a need to recommend top-K items to new users who have joined recently. Since the joined, users have not rated any items. Thus, there is a need to generate item recommendations that are based on static similarities. Therefore, the weights ω_C and ω_P were set to 0, while the weights of ω_A and ω_O were set to 0.5.

Table 7-5: Measured MAE values for scenario 2.

K	Precision									
	K-Means		FCM		AP		WNMF		Proposed	
	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng	MovieLens	Ta-Feng
5	2.63	2.74	2.61	2.78	2.56	2.65	2.85	2.8	1.63	1.59
10	2.72	2.76	2.76	2.84	2.61	2.64	3.18	3.1	1.65	1.62
20	2.85	2.8	2.83	2.76	2.87	2.93	3.19	3.22	1.72	1.68
30	2.98	3.1	2.97	2.83	2.92	2.89	3.32	3.4	1.78	1.72
40	3.13	3.26	3.14	3.15	3.1	3.08	3.38	3.41	1.85	1.8
50	3.34	3.4	3.42	3.23	3.22	3.27	3.45	3.38	1.92	1.83

To show the cold start performance of the proposed approach only MAE measure is shown (Table 7-6), because objective of this scenario was to validate its superiority and the detailed comparison was already performed in scenario 1. The MAE for cold-start conditions is shown in Table 7-6 which shows that the proposed approach achieves 40.37% and 43.34% improvements for MovieLens1M and Ta-Feng datasets, respectively.

7.3.3 Scenario 3

In this scenario, the algorithms were evaluated to test their performance during temporal variations in the user's behavior. To evaluate this performance, the deviation in MAE during successive segments of time is tabulated in Tables 7 and 8 for the MovieLens1M and Ta-Feng datasets, respectively. These results show that the proposed approach achieves the lowest standard deviations of 0.153 and 0.131 followed by 0.296 and 0.239 by AP and FCM for MovieLens1M

Table 7-6: Measured MAE values for MovieLens dataset at successive time segments for scenario 3.

K	Time Segment									
	t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9
K-Means	2.63	2.51	2.35	2.02	1.78	1.75	1.96	2.21	1.71	1.7
FCM	2.61	2.55	2.25	2.11	1.81	1.8	2.11	2.25	1.8	1.75
AP	2.56	2.49	2.15	1.98	1.75	1.91	2.05	2.35	1.82	1.65
WNMF	2.85	2.65	2.45	1.73	1.82	1.89	2.15	2.22	1.92	1.67
Proposed	1.63	1.54	1.24	1.15	1.18	1.21	1.22	1.36	1.26	1.2
K-Means	2.63	2.51	2.35	2.02	1.78	1.75	1.96	2.21	1.71	1.7

Table 7-7: Measured MAE values for Ta-Feng dataset at successive time segments for scenario 3.

K	Time Segment									
	t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9
K-Means	2.55	2.45	2.21	1.96	1.7	1.78	1.86	2.26	1.81	1.75
FCM	2.43	2.35	2.3	1.95	1.72	1.88	2.23	2.2	1.85	1.85
AP	2.32	2.45	2.25	1.84	1.65	1.81	2.15	2.25	1.95	1.75
WNMF	2.77	2.61	2.17	1.63	1.52	1.69	2.25	2.28	1.82	1.87
Proposed	1.59	1.51	1.24	1.25	1.19	1.24	1.23	1.26	1.32	1.24
K-Means	2.55	2.45	2.21	1.96	1.7	1.78	1.86	2.26	1.81	1.75

and Ta-Feng datasets, respectively. These results demonstrate that the proposed algorithm provides consistent, steady performance with a minimum deviation from a cold start (at t0) to all other segments. This validates the impact of static and temporal similarities on managing steady performance. The weights ω_C, ω_A, ω_O and ω_P are set to 0.5, 0.15, 0.15, and 0.2.

7.3.4 Scenario 4

This scenario was developed to compare the proposed algorithm with some recent algorithms discussed in [23] named NeuMF, Wide & Deep, deepCF, lCoupledCF, gCoupledCF, and CoupledCF. For this scenario the weights ω_C, ω_A, ω_O and ω_P are set to 0.5, 0.15, 0.15, and 0.2.

Table 7-8: HR@K (K = 10) performance comparison for scenario 4.

Dataset	HR@10						
	NeuMF	Wide & deep	deepCF	lCoupledCF	gCoupledCF	CoupledCF	Proposed
MovieLens	2.63	2.51	2.35	2.02	1.78	1.75	1.96
Ta-Feng	2.61	2.55	2.25	2.11	1.81	1.8	2.11

Table 7-9: NDCG@K (K = 10) performance comparison for scenario 4.

Dataset	NDCG@10						
	NeuMF	Wide & deep	deepCF	lCoupledCF	gCoupledCF	CoupledCF	Proposed
MovieLens	0.448	0.447	0.4312	0.5408	0.5252	0.544	0.5489
Ta-Feng	0.4329	0.4233	0.4322	0.47	0.4205	0.4623	0.457

As shown in Tables 7-9 and 7-10, our proposed technique outperforms the neural MF model NeuMF by 15.21% on MovieLens1M and 9.57% on the Ta-Feng dataset for HR@10 and NDCG@10 respectively. Compared to the MF's neural model, ours outperforms it by a wide margin. According to MovieLens1M's HR@10 and NDCG@10 results, the suggested method exceeded the Google benchmark Wide & Deep model by a total of 22.52% and 11.26%, respectively, compared to Google's Wide & Deep model. This demonstrates how the proposed approach, which incorporates static and temporal similarities, outperforms the Wide & Deep's based on feature engineering. It also outperforms its closest competitor, CoupledCF, by 2.06% and 2.75% for MovieLens1M and Ta-Feng datasets, respectively, on HR@10.

7.4 Conclusion

In this section, a collaborative filtering-based recommender system has been presented. The proposed approach involves both static (non-temporal) and dynamic (temporal) characteristics for user similarity estimation and adopts model-based and graph-based approaches for rating estimation. The use of affinity clustering in a model-based approach greatly reduces the user-item matrix size, which results in lower memory and processing requirements. Furthermore, the involvement of both static and dynamic characteristics not only helps in gathering more information but also makes the system more flexible. The evaluation results also show that the proposed algorithm significantly improves the overall performance of the recommender system under normal conditions, cold start conditions, and provides consistent steady performance during temporal variations. However, in the presented work, the final ratings are estimated by a weighted aggregation of the ratings achieved by different techniques independently. In the process, the weights are manually selected, which may not produce the optimum results. An optimal weight selection algorithm can be developed to overcome this limitation. However, this work is left for future improvements.

CHAPTER 8
LINK PREDICTION BASED ON BIPARTITE GRAPH FOR RECOMMENDATION SYSTEM USING OPTIMIZED SVD++

8.1 Introduction

There is a need to filter, prioritize, and effectively transmission of the important information on the Internet, where the variety of options is overwhelming, in order to ease the big data challenges, which has created a possible issue for many Internet users. This challenge is solved by recommender systems, which search through a massive amount of dynamically created data to present users with tailored content and services. Because of the sparse nature of this data, bipartite graphs are nowadays widely opted to store and interpret it. A recommendation problem is transformed into a link prediction problem by mapping data to a bipartite user-item interaction graph, where the graph topology captures nuanced information about user-item relationships. Earlier, approaches for link prediction in bipartite graphs for various recommendation systems were developed, but the efficacy of the prediction methodology was not close to the standards required by real-time recommenders. So, the primary goal of this research is to offer an effective link prediction-based recommendation system that takes advantage of SVD++ along with K-nearest neighbors and reduces the system's error rate, leading to better outcomes. The proposed system is sorely tested on the MovieLens dataset and compared to some traditional recommendation methods. The findings show that the proposed approach outperforms all conventional methods in terms of accuracy, and the actual recommendations are also promising.

8.2 Proposed Link prediction-based Recommendation System

The proposed recommendation system is designed for movie recommendation, and it uses the concept of link prediction-based recommendations. Here matrix factorization is used for link prediction where SVD++ algorithm is utilized and to optimize the outcome KNN algorithm is opted. Both are machine learning approaches and provides an effective result as required from the system. The proposed approach is compared with the other content, collaborative filtering, and

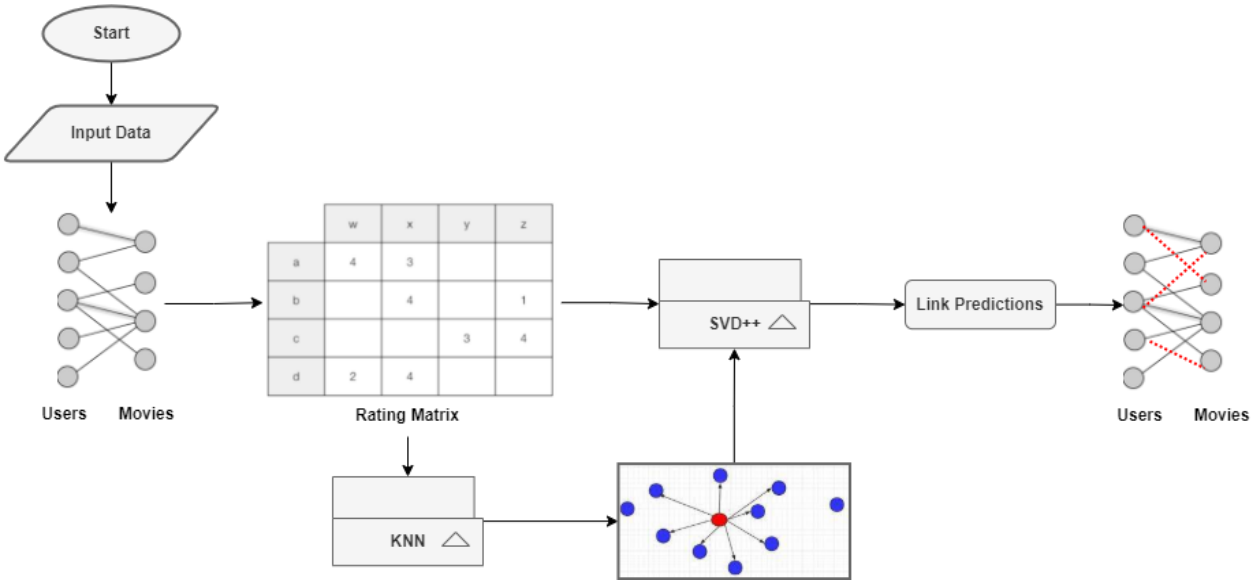


Figure 8.1: Proposed Link Prediction based Recommendation System.

link prediction-based approaches. The proposed model of the system can be depicted from the Fig. 8.1.

The singular value decomposition plus plus (SVD++) model, in comparison to the classic MF technique, includes user bias information and implicit parameters to objectively characterize user preferences. User ratings indicate personal preferences; some ratings are poor (or high). Therefore, the proposed model employs SVD++ to better mine user preferences and produce expected ratings with more accuracy. In the SVD++ model, the rating prediction calculation is given by Equation.

$$L_{(u,i)} = b_u + b_i + m + F_i^T \left(F_u + |I_u|^{-\frac{1}{2}} \sum_{n \in I_u} M_n \right) \tag{8.1}$$

where $L_{(u,i)}$ is the anticipated rating provided by user *'u'* to item *'i'* and '*m'* represents the global ratings mean. b_u stands for user bias which is independent of the item's qualities. Item bias is represented by b_i, and it is independent of user interest. The mean of the worldwide evaluations is thus represented by '*m*'. The latent feature vectors of object *'i'* and user *'u'* are referred to as 'F_i' and 'F_u', respectively. The set of things that generate implicit feedback for a certain user '*u'* is defined as 'I_u'. The implicit effect of items rated by user '*u'* in the past on the ratings of unknown objects in the future is denoted by *'M_n'*, and it includes all of the items rated by user *u* in the past. All of user *u's* ratings revised values are $\sum_{n \in I_u} M_n$, where M_n signifies the implicit impact of objects rated by user u in the past on the ratings of unknown items in the future. These are commonly written as $F_i^T M_n$, with the addition of $|I_u|^{-\frac{1}{2}}$ to account for differences in I_u numbers.

To summarise, the SVD++ model represents user preferences using bias data, implicit parameters, and other factors.

To optimize the prediction of SVD++, K-nearest neighbour algorithm is used here in this work. The link prediction in KNN is performed based on the distance-based measure that predict the links between user *'u'* and item *'i'*. The following equation computes the optimized links:

$$L'_{(u,i)} = L^{k}_{(u,i)} \cap \left[b_u + b_i + m + F_i^T \left(F_u + |I_u|^{-\frac{1}{2}} \sum_{n \in I_u} M_n \right) \right] \quad (8.2)$$

Here, $L^{k}_{(u,i)}$ are the predicted links using KNN method. The pseudo code of the proposed hybrid link prediction algorithm is as given below:

Algorithm 1: Optimized SVD++ based Link Prediction
Input: *G= (u, i, E, W_{ui}) //user 'u', item 'i', edge 'E', weight 'W_{ui}'*
Output: $\boldsymbol{L'_{(u,i)}}$ //Predicted Links between user and items
Start
Input the given data and generate bipartite graph
Convert the graph nodes into matrix with given weight 'W_{ui}' values
Calculate mean of the weight values 'm'
Initialize the bias information b_u and b_i. Initialize the user vector F_u and the item vector F_i. Initialize the implicit parameters M_n.
Compute $\boldsymbol{L_{(u,i)}}$ using equation (i)
Compute Intersection between the links predicted using KNN '$\boldsymbol{L^{k}_{(u,i)}}$', and $\boldsymbol{L_{(u,i)}}$ using equation (ii)
Repeat the last two step to predict the new links between *u, i.*
Stop

8.3 Experimentation and Result Analysis

The proposed approach is implemented and compared with the traditional random, content, collaborative filtering, KNN, SVD, SVD++ using MovieLens dataset.

Table 8-1: Experimentation Results

Methods	**RMSE**	**MAE**	**HR**	**cHR**	**ARHR**	**Coverage**
Random	1.4385	1.1478	0.0089	0.0089	0.0015	1.0000
Content based	0.9375	0.7263	0.0030	0.0030	0.0017	0.9285
Collaborative Filtering (User based)	0.9961	0.7711	0	0	0	1
Collaborative Filtering (Item based)	0.9995	0.7798	0	0	0	0.9896
Link Prediction using KNN	0.9321	0.7210	0.02718	0.02718	0.0110	0.9230

Link Prediction using SVD	0.9039	0.6984	0.0283	0.0283	0.0119	0.9478
Link Prediction using SVD++	0.8928	0.6865	0.0328	0.0328	0.0143	0.9434
Link Prediction using Optimized SVD++	0.8123	0.6524	0.04126	0.04126	0.0156	1

The results in Table 8-1 shows that the RMSE of the proposed link prediction approach is lesser than all other traditional approaches. As per analysis it is 9%, 10.1%, and 12.8% better from the other link prediction algorithms SVD++, SVD, and KNN respectively. Moreover, the proposed approach is 18% better than the collaborative filtering and 8.7% better than the Content based recommendation systems. The above results also shows that the MAE of the proposed link prediction approach is lesser than all other traditional approaches. As per analysis it is 4.9%, 6.5%, and 9.5% better from the other link prediction algorithms SVD++, SVD, and KNN respectively. Moreover, the proposed approach is 15% better than the collaborative filtering and 6.4% better than the Content based recommendation systems.

8.4 Conclusion

The proposed Link prediction-based recommendation system is more efficient than standard recommendation systems, according to this research. Because it is based on the local vicinity of a huge sparse network, and the link prediction technique is scalable. The information supplied on the link is not efficiently utilized by traditional recommender systems. The link prediction in this research is done with a bipartite network, and it is based on matrix factorization. The final linkages between users and items are predicted using the SVD++, which are subsequently optimized using the KNN technique. The findings show that the proposed technique outperforms existing traditional recommendation systems and link prediction approaches in terms of different performance measures. The great accuracy attained by the recommender system was indicated by the error rate in terms of RMSE and MAE. Other performance metrics, such as hit rate, ARHR, Coverage, and Diversity, also demonstrate the usefulness of the proposed method and show promising outcomes. Deep learning technologies may be evaluated on other sets of data in the future to increase the system's accuracy.

CHAPTER 9
CONCLUSION

Collaborative filtering is used in recommendation systems. In CF, similar users' interests and preferences are analyzed to recommend items. Existing methods calculate the correlation coefficient between users, which cannot measure sequential and temporal similarity, leading to poor performance. We proposed three graph-based approaches to manage these problems.

Firstly, the findings for the third approach indicate that the proposed technique outperforms existing traditional recommendation systems and link prediction approaches on a variety of performance metrics. The RMSE and MAE error rates suggested that the recommender system achieved a high level of accuracy. Other performance indicators, such as hit rate, ARHR, Coverage, and Diversity, also highlight the use and promise of the suggested technique.

Second, a proposed technique defines user similarity using Hidden Markov Model (HMM) parameters to extract key information. The method has the following features: (i) a temporal graph-based algorithm extracts information, (ii) similarity-based weighted averages make the system more resilient to errors in weight estimation, resulting in better information collaboration, and (iii) the method reduces similarity measurement complexity compared to graph-based algorithms that have an exponentially increasing number of branches. These features improve overall performance based on various evaluation metrics.

Third, a recommender system based on graph and collaborative filtering has been presented. For user similarity estimation, the suggested method employs both static (non-temporal) and dynamic (temporal) attributes and model-based and graph-based approaches for rating estimation. Using affinity clustering in a model-based method drastically reduces the size of the user-item matrix, resulting in reduced memory and processing needs. Moreover, the inclusion of both static and dynamic attributes not only facilitates the collection of more data, but also increases the system's adaptability. In addition, the assessment findings demonstrate that the suggested algorithm considerably enhances the overall performance of the recommender system under normal conditions, cold start conditions, and during temporal changes.

CHAPTER 10
FUTURE SCOPE

Although the proposed algorithms perform well in all experimental analysis, in future following improvements can be added to further improve the performance:

1. In the presented work, the final ratings are estimated by a weighted aggregation of the ratings achieved by different techniques independently. In the process, the weights are manually selected, which may not produce the optimum results. An optimal weight selection algorithm can be developed to overcome this limitation.
2. Deep learning technologies may be evaluated on other sets of data in the future to increase the system's accuracy.

CHAPTER 11
LIST OF PUBLICATIONS

- Gupta, A., & Srinath, P. (2021). Survey on Recommender System using Hybrid Collaborative Filtering Techniques in Online Social Network. Design Engineering, 14468-14479.

- Gupta, A., & Srinath, P. (2022). A recommender system based on collaborative filtering, graph theory using HMM based similarity measures. International Journal of System Assurance Engineering and Management, 1-13.

- Gupta, A., & Shrinath, P. (2022). A novel recommendation system comprising WNMF with graph-based static and temporal similarity estimators. International Journal of Data Science and Analytics, 1-15.

- Gupta, A., & Shrinath, P. (2022). Link Prediction based on Bipartite graph for Recommendation System using optimized SVD++, International Conference on Machine Learning and Data Engineering, Procedia Computer Science Journal, Elsevier.

REFERENCES

[1] B. Shams and S. Haratizadeh, "Graph-based collaborative ranking," *Expert Systems with Applications*, vol. 67, pp. 59–70, Jan. 2017, doi: 10.1016/j.eswa.2016.09.013.

[2] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowledge-Based Systems*, vol. 46, pp. 109–132, Jul. 2013, doi: 10.1016/j.knosys.2013.03.012.

[3] F. Ricci, L. Rokach, and B. Shapira, "Recommender Systems: Introduction and Challenges," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. Boston, MA: Springer US, 2015, pp. 1–34. doi: 10.1007/978-1-4899-7637-6_1.

[4] C. C. Aggarwal, "An Introduction to Recommender Systems," in *Recommender Systems*, Cham: Springer International Publishing, 2016, pp. 1–28. doi: 10.1007/978-3-319-29659-3_1.

[5] R. Burke, "Hybrid Web Recommender Systems," in *The Adaptive Web: Methods and Strategies of Web Personalization*, P. Brusilovsky, A. Kobsa, and W. Nejdl, Eds. Berlin, Heidelberg: Springer, 2007, pp. 377–408. doi: 10.1007/978-3-540-72079-9_12.

[6] S. M. Ali, G. K. Nayak, R. K. Lenka, and R. K. Barik, "Movie Recommendation System Using Genome Tags and Content-Based Filtering," in *Advances in Data and Information Sciences*, Singapore, 2018, pp. 85–94. doi: 10.1007/978-981-10-8360-0_8.

[7] D. Kluver, M. D. Ekstrand, and J. A. Konstan, "Rating-Based Collaborative Filtering: Algorithms and Evaluation," in *Social Information Access: Systems and Technologies*, P. Brusilovsky and D. He, Eds. Cham: Springer International Publishing, 2018, pp. 344–390. doi: 10.1007/978-3-319-90092-6_10.

[8] I. Elachkar, H. Ouzif, and H. Labriji, "STRUCTURAL SIMILARITY MEASURE OF USERS PROFILES BASED ON A WEIGHTED BIPARTITE GRAPHS," in *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Nov. 2020, vol. XLIV-4-W3-2020, pp. 203–207. doi: 10.5194/isprs-archives-XLIV-4-W3-2020-203-2020.

[9] A. K. Pandey and D. S. Rajpoot, "Resolving Cold Start problem in recommendation system using demographic approach," in *2016 International Conference on Signal Processing and Communication (ICSC)*, Dec. 2016, pp. 213–218. doi: 10.1109/ICSPCom.2016.7980578.

[10] M. Göker and L. McGinty, "Case-based recommender systems," *The Knowledge Engineering Review*, Jan. 2005, Accessed: May 20, 2022. [Online]. Available: https://www.academia.edu/16858234/Case_based_recommender_systems

[11] I. Viktoratos, A. Tsadiras, and N. Bassiliades, "Combining community-based knowledge with association rule mining to alleviate the cold start problem in context-aware recommender systems," *Expert Systems with Applications*, vol. 101, pp. 78–90, Jul. 2018, doi: 10.1016/j.eswa.2018.01.044.

[12] E. Çano and M. Morisio, "Hybrid Recommender Systems: A Systematic Literature Review," *IDA*, vol. 21, no. 6, pp. 1487–1524, Nov. 2017, doi: 10.3233/IDA-163209.

[13] Y. Shi, M. Larson, and A. Hanjalic, "Unifying rating-oriented and ranking-oriented collaborative filtering for improved recommendation," *Information Sciences*, vol. 229, pp. 29–39, Apr. 2013, doi: 10.1016/j.ins.2012.12.002.

[14] S. Goldberg, M. Schapira, P. Hummon, and J. Rexford, "How secure are secure interdomain routing protocols," *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 4, pp. 87–98, Aug. 2010, doi: 10.1145/1851275.1851195.

[15] Z. Wang, X. Yu, N. Feng, and Z. Wang, "An improved collaborative movie recommendation system using computational intelligence," *Journal of Visual Languages & Computing*, vol. 25, no. 6, pp. 667–675, Dec. 2014, doi: 10.1016/j.jvlc.2014.09.011.

[16] N. Mustafa, A. O. Ibrahim, A. Ahmed, and A. Abdullah, "Collaborative filtering: Techniques and applications," in *2017 International Conference on Communication, Control, Computing and Electronics Engineering (ICCCCEE)*, Jan. 2017, pp. 1–6. doi: 10.1109/ICCCCEE.2017.7867668.

[17] D.-K. Chae, S.-C. Lee, S.-Y. Lee, and S.-W. Kim, "On identifying k-nearest neighbors in neighborhood models for efficient and effective collaborative filtering," *Neurocomputing*, vol. 278, pp. 134–143, Feb. 2018, doi: 10.1016/j.neucom.2017.06.081.

[18] H. Koohi and K. Kiani, "User based Collaborative Filtering using fuzzy C-means," *Measurement*, vol. C, no. 91, pp. 134–139, 2016, doi: 10.1016/j.measurement.2016.05.058.

[19] P. H. Aditya, I. Budi, and Q. Munajat, "A comparative analysis of memory-based and model-based collaborative filtering on the implementation of recommender system for E-commerce in Indonesia: A case study PT X," in

2016 International Conference on Advanced Computer Science and Information Systems (ICACSIS), Oct. 2016, pp. 303–308. doi: 10.1109/ICACSIS.2016.7872755.

[20] S. Wang *et al.*, "Hierarchical Attentive Transaction Embedding With Intra- and Inter-Transaction Dependencies for Next-Item Recommendation," *IEEE Intelligent Systems*, vol. 36, no. 4, pp. 56–64, Jul. 2021, doi: 10.1109/MIS.2020.2997362.

[21] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep Learning based Recommender System: A Survey and New Perspectives," *ACM Comput. Surv.*, vol. 52, no. 1, pp. 1–38, Jan. 2020, doi: 10.1145/3285029.

[22] W. Lu, Y. Yu, Y. Chang, Z. Wang, C. Li, and B. Yuan, "A Dual Input-aware Factorization Machine for CTR Prediction," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, Yokohama, Japan, Jul. 2020, pp. 3139–3145. doi: 10.24963/ijcai.2020/434.

[23] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "DeepFM: A Factorization-Machine based Neural Network for CTR Prediction," *arXiv:1703.04247 [cs]*, Mar. 2017, Accessed: May 07, 2022. [Online]. Available: http://arxiv.org/abs/1703.04247

[24] H.-T. Cheng *et al.*, "Wide & Deep Learning for Recommender Systems," in *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, New York, NY, USA, Sep. 2016, pp. 7–10. doi: 10.1145/2988450.2988454.

[25] Q. Zhang, L. Cao, C. Zhu, Z. Li, and J. Sun, "CoupledCF: Learning Explicit and Implicit User-item Couplings in Recommendation for Deep Collaborative Filtering," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, Jul. 2018, pp. 3662–3668. doi: 10.24963/ijcai.2018/509.

[26] Y. Tay, A. T. Luu, and S. C. Hui, "Latent Relational Metric Learning via Memory-based Attention for Collaborative Ranking," *Proceedings of the 2018 World Wide Web Conference on World Wide Web - WWW '18*, pp. 729–739, 2018, doi: 10.1145/3178876.3186154.

[27] Y. Zhang, G. Bai, M. Zhong, X. Li, and R. K. L. Ko, "Differentially Private Collaborative Coupling Learning for Recommender Systems," *IEEE Intelligent Systems*, vol. 36, no. 1, pp. 16–24, Jan. 2021, doi: 10.1109/MIS.2020.3005930.

[28] Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009, doi: 10.1109/MC.2009.263.

[29] R. Mehta and K. Rana, "A review on matrix factorization techniques in recommender systems," in *2017 2nd International Conference on Communication Systems, Computing and IT Applications (CSCITA)*, Apr. 2017, pp. 269–274. doi: 10.1109/CSCITA.2017.8066567.

[30] M. Sunitha Reddy and T. Adilakshmi, "Music recommendation system based on matrix factorization technique -SVD," in *2014 International Conference on Computer Communication and Informatics*, Jan. 2014, pp. 1–6. doi: 10.1109/ICCCI.2014.6921744.

[31] H. Yi, Z. Niu, F. Zhang, X. Li, and Y. Wang, "Robust Recommendation Algorithm Based on Kernel Principal Component Analysis and Fuzzy C-means Clustering," *Wuhan Univ. J. Nat. Sci.*, vol. 23, no. 2, pp. 111–119, Apr. 2018, doi: 10.1007/s11859-018-1301-6.

[32] H. Ma, H. Yang, M. R. Lyu, and I. King, "SoRec: social recommendation using probabilistic matrix factorization," in *Proceedings of the 17th ACM conference on Information and knowledge management*, New York, NY, USA, Oct. 2008, pp. 931–940. doi: 10.1145/1458082.1458205.

[33] J. Liu, C. Wu, and W. Liu, "Bayesian Probabilistic Matrix Factorization with Social Relations and Item Contents for recommendation," *Decision Support Systems*, vol. 55, no. 3, pp. 838–850, Jun. 2013, doi: 10.1016/j.dss.2013.04.002.

[34] X. Luo, M. Zhou, Y. Xia, and Q. Zhu, "An Efficient Non-Negative Matrix-Factorization-Based Approach to Collaborative Filtering for Recommender Systems," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1273–1284, May 2014, doi: 10.1109/TII.2014.2308433.

[35] G. Dror, N. Koenigstein, Y. Koren, and M. Weimer, "The Yahoo! Music Dataset and KDD-Cup'11," in *Proceedings of KDD Cup 2011*, Jun. 2012, pp. 3–18. Accessed: May 07, 2022. [Online]. Available: https://proceedings.mlr.press/v18/dror12a.html

[36] G. Takács and D. Tikk, "Alternating least squares for personalized ranking," in *Proceedings of the sixth ACM conference on Recommender systems*, New York, NY, USA, Sep. 2012, pp. 83–90. doi: 10.1145/2365952.2365972.

[37] L. Bottou, "Stochastic Gradient Descent Tricks," in *Neural Networks: Tricks of the Trade: Second Edition*, G. Montavon, G. B. Orr, and K.-R. Müller, Eds. Berlin, Heidelberg: Springer, 2012, pp. 421–436. doi: 10.1007/978-3-642-35289-8_25.

[38] H.-F. Yu, C.-J. Hsieh, S. Si, and I. S. Dhillon, "Parallel matrix factorization for recommender systems," *Knowl Inf Syst*, vol. 41, no. 3, pp. 793–819, Dec. 2014, doi: 10.1007/s10115-013-0682-2.
[39] Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan, "Large-Scale Parallel Collaborative Filtering for the Netflix Prize," in *Algorithmic Aspects in Information and Management*, Berlin, Heidelberg, 2008, pp. 337–348. doi: 10.1007/978-3-540-68880-8_32.
[40] L. Feng, H. Wei, Q. Guo, Z. Lin, and B. An, "Embedding-Augmented Generalized Matrix Factorization for Recommendation With Implicit Feedback," *IEEE Intelligent Systems*, vol. 36, no. 6, pp. 32–41, Nov. 2021, doi: 10.1109/MIS.2020.3036136.
[41] F. Li, G. Xu, and L. Cao, "Coupled Matrix Factorization within Non-IID Context," *arXiv:1404.7467 [cs]*, Dec. 2014, Accessed: May 07, 2022. [Online]. Available: http://arxiv.org/abs/1404.7467
[42] H. J. Ahn, "A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem," *Inf. Sci.*, vol. 178, no. 1, pp. 37–51, Jan. 2008, doi: 10.1016/j.ins.2007.07.024.
[43] Suryakant and T. Mahara, "A New Similarity Measure Based on Mean Measure of Divergence for Collaborative Filtering in Sparse Environment," *Procedia Computer Science*, vol. 89, pp. 450–456, 2016, doi: 10.1016/j.procs.2016.06.099.
[44] J. Bobadilla, F. Serradilla, and J. Bernal, "A new collaborative filtering metric that improves the behavior of recommender systems," *Knowledge-Based Systems*, vol. 23, no. 6, pp. 520–528, Aug. 2010, doi: 10.1016/j.knosys.2010.03.009.
[45] J. Bobadilla, F. Ortega, A. Hernando, and J. Bernal, "A collaborative filtering approach to mitigate the new user cold start problem," *Knowledge-Based Systems*, vol. 26, pp. 225–238, Feb. 2012, doi: 10.1016/j.knosys.2011.07.021.
[46] K. Choi and Y. Suh, "A new similarity function for selecting neighbors for each target item in collaborative filtering," *Knowledge-Based Systems*, vol. 37, pp. 146–153, Jan. 2013, doi: 10.1016/j.knosys.2012.07.019.
[47] B. Jeong, J. Lee, and H. Cho, "Improving memory-based collaborative filtering via similarity updating and prediction modulation," *Information Sciences*, vol. 180, no. 5, pp. 602–612, Mar. 2010, doi: 10.1016/j.ins.2009.10.016.
[48] W. Fan *et al.*, "Graph Neural Networks for Social Recommendation," *arXiv:1902.07243 [cs]*, Nov. 2019, Accessed: May 07, 2022. [Online]. Available: http://arxiv.org/abs/1902.07243
[49] X. Li and H. Chen, "Recommendation as link prediction in bipartite graphs: A graph kernel-based machine learning approach," *Decision Support Systems*, vol. 54, no. 2, pp. 880–890, Jan. 2013, doi: 10.1016/j.dss.2012.09.019.
[50] E. Palumbo, G. Rizzo, and R. Troncy, "entity2rec: Learning User-Item Relatedness from Knowledge Graphs for Top-N Item Recommendation," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, Como Italy, Aug. 2017, pp. 32–36. doi: 10.1145/3109859.3109889.
[51] S. Wang *et al.*, "Graph Learning Approaches to Recommender Systems: A Review," p. 8.
[52] J. Verma, S. Gupta, D. Mukherjee, and T. Chakraborty, "Heterogeneous Edge Embeddings for Friend Recommendation," *arXiv:1902.03124 [cs, stat]*, Feb. 2019, Accessed: May 07, 2022. [Online]. Available: http://arxiv.org/abs/1902.03124
[53] Y. Cen, X. Zou, J. Zhang, H. Yang, J. Zhou, and J. Tang, "Representation Learning for Attributed Multiplex Heterogeneous Network," *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1358–1368, Jul. 2019, doi: 10.1145/3292500.3330964.
[54] Q. Guo *et al.*, "A Survey on Knowledge Graph-Based Recommender Systems," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2020, doi: 10.1109/TKDE.2020.3028705.
[55] J. Cheng, Y. Liu, H. Zhang, X. Wu, and F. Chen, "A New Recommendation Algorithm Based on User's Dynamic Information in Complex Social Network," *Mathematical Problems in Engineering*, vol. 2015, p. e281629, Mar. 2015, doi: 10.1155/2015/281629.
[56] Q. Tan and F. Liu, "Recommendation Based on Users' Long-Term and Short-Term Interests with Attention," *Mathematical Problems in Engineering*, vol. 2019, p. e7586589, Oct. 2019, doi: 10.1155/2019/7586589.
[57] O. C. Ibe, "3 - Introduction to Markov Processes," in *Markov Processes for Stochastic Modeling (Second Edition)*, O. C. Ibe, Ed. Oxford: Elsevier, 2013, pp. 49–57. doi: 10.1016/B978-0-12-407795-9.00003-7.
[58] M. Franzese and A. Iuliano, "Hidden Markov Models," in *Encyclopedia of Bioinformatics and Computational Biology*, S. Ranganathan, M. Gribskov, K. Nakai, and C. Schönbach, Eds. Oxford: Academic Press, 2019, pp. 753–762. doi: 10.1016/B978-0-12-809633-8.20488-3.
[59] O. C. Ibe, "Chapter 12 - Special Random Processes," in *Fundamentals of Applied Probability and Random Processes (Second Edition)*, O. C. Ibe, Ed. Boston: Academic Press, 2014, pp. 369–425. doi: 10.1016/B978-0-12-800852-2.00012-2.

[60] A. Zahari and J. Jaafar, “A novel approach of hidden Markov model for time series forecasting,” in *Proceedings of the 9th International Conference on Ubiquitous Information Management and Communication*, New York, NY, USA, Jan. 2015, pp. 1–5. doi: 10.1145/2701126.2701179.
[61] A. Zeifman, V. Korolev, and A. Sipin, *Stochastic Processes: Theory and Applications*. MDPI, 2019.
[62] L. R. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989, doi: 10.1109/5.18626.
[63] S. Theodoridis, “Chapter 12 - Bayesian Learning: Inference and the EM Algorithm,” in *Machine Learning (Second Edition)*, S. Theodoridis, Ed. Academic Press, 2020, pp. 595–646. doi: 10.1016/B978-0-12-818803-3.00023-4.
[64] C. Zhai, “A Brief Note on the Hidden Markov Models (HMM.” 2003.
[65] Y. Li and A. Ngom, “The non-negative matrix factorization toolbox for biological data mining,” *Source Code for Biology and Medicine*, vol. 8, no. 1, p. 10, Apr. 2013, doi: 10.1186/1751-0473-8-10.